



BÁCH KHOA E-LEARNING

[Trang của tôi](#) / [Khoá học](#) / [Học kỳ II năm học 2021-2022 \(Semester 2 - Academic year 2021-2022\)](#).

/ [Đại Học Chính Quy \(Bachelor program \(Full-time study\)\)](#). / [Khoa Khoa học và Kỹ thuật Máy tính \(Faculty of Computer Science and Engineering\)](#).

/ [Khoa Học Máy Tính](#) / [Nguyên lý ngôn ngữ lập trình \(CO3005\) - Trần Ngọc Bảo Duy \(DH_HK212\)](#).

/ [Cây cú pháp trừu tượng - Abstract Syntax Tree \(Buổi 4\)](#) / [Programming Code: AST \(extra exercise\)](#).

Câu hỏi 1

Không hoàn thành

Chấm điểm của 1,00

In CSEL, a program consists of many declarations: variable declaration (vardecl), constant declaration (constdecl), function declaration (funcdecl). Given the grammar of CSEL as follows:

```
program: decl+ EOF;
cseltype: INT | FLOAT | BOOLEAN;
decl: vardecl decltail | constdecl decltail | funcdecl decltail;
decltail: vardecl decltail | constdecl decltail | funcdecl decltail | ;
vardecl: LET single_vardecls SEMI;
single_vardecls: single_vardecl single_vardecltail;
single_vardecl: ID COLON cseltype;
single_vardecltail: COMMA single_vardecl single_vardecltail | ;
constdecl: CONST single_constdecl SEMI;
single_constdecl: ID COLON cseltype EQ expr;
expr: INTLIT | FLOATLIT | BOOLEANLIT;
funcdecl: FUNCTION ID LR paramlist RR SEMI;
paramlist: single_vardecls | ;
LET: 'Let';
CONST: 'Constant';
FUNCTION: 'Function';
SEMI: ';';
COLON: ':';
COMMA: ',';
LR: '(';
RR: ')';
```

```

EQ: '=';
INT: 'Int';
FLOAT: 'Float';
BOOLEAN: 'Boolean';
INTLIT: [0-9]+;
FLOATLIT: [0-9]+ '.' [0-9]+;
BOOLEANLIT: 'True' | 'False';
ID: [a-zA-Z]+;
WS: [ \t\r\n\f]+ -> skip;

```

and AST classes as follows:

```

class Program(ABC): # decl: List[Decl]

class Type(ABC): pass

class IntType(Type)

class FloatType(Type)

class BooleanType(Type)

class LHS(ABC): pass

class Id(LHS): # name: str

class Decl(ABC): pass

class VarDecl(Decl): # id: Id, typ: Type

class ConstDecl(Decl): # id: Id, typ: Type, value: Expr

class FuncDecl(Decl): # name: Id, param: List[VarDecl]

class Exp(ABC): pass

class IntLit(Exp): # value: int

class FloatLit(Exp): # value: float

```

```
class BooleanLit(Exp): # value: bool
```

Please copy the following class into your answer and modify the bodies of its methods to generate the AST of a CSEL input?

```
class ASTGenerator(CSELVisitor):
```

```
    # Visit a parse tree produced by CSELParse#program.
```

```
    def visitProgram(self, ctx:CSELParse.ProgramContext):
```

```
        return self.visitChildren(ctx)
```

```
    # Visit a parse tree produced by CSELParse#cseltype.
```

```
    def visitCseltype(self, ctx:CSELParse.CseltypeContext):
```

```
        return self.visitChildren(ctx)
```

```
    # Visit a parse tree produced by CSELParse#decl.
```

```
    def visitDecl(self, ctx:CSELParse.DeclContext):
```

```
        return self.visitChildren(ctx)
```

```
    # Visit a parse tree produced by CSELParse#decltail.
```

```
    def visitDecltail(self, ctx:CSELParse.DecltailContext):
```

```
        return self.visitChildren(ctx)
```

```
    # Visit a parse tree produced by CSELParse#vardecl.
```

```
    def visitVardecl(self, ctx:CSELParse.VardeclContext):
```

```
        return self.visitChildren(ctx)
```

```
    # Visit a parse tree produced by CSELParse#single_vardecls.
```

```
    def visitSingle_vardecls(self, ctx:CSELParse.Single_vardeclsContext):
```

```
        return self.visitChildren(ctx)
```

```
    # Visit a parse tree produced by CSELParse#single_vardecl.
```

```
    def visitSingle_vardecl(self, ctx:CSELParse.Single_vardeclContext):
```

```
        return self.visitChildren(ctx)
```

Thời gian còn lại 0:42:23

```

# Visit a parse tree produced by CSELParse#single_vardecltail.
def visitSingle_vardecltail(self, ctx:CSELParse.Single_vardecltailContext):
    return self.visitChildren(ctx)

# Visit a parse tree produced by CSELParse#constdecl.
def visitConstdecl(self, ctx:CSELParse.ConstdeclContext):
    return self.visitChildren(ctx)

# Visit a parse tree produced by CSELParse#single_constdecl.
def visitSingle_constdecl(self, ctx:CSELParse.Single_constdeclContext):
    return self.visitChildren(ctx)

# Visit a parse tree produced by CSELParse#expr.
def visitExpr(self, ctx:CSELParse.ExprContext):
    return self.visitChildren(ctx)

# Visit a parse tree produced by CSELParse#funcdecl.
def visitFuncdecl(self, ctx:CSELParse.FuncdeclContext):
    return self.visitChildren(ctx)

# Visit a parse tree produced by CSELParse#paramlist.
def visitParamlist(self, ctx:CSELParse.ParamlistContext):
    return self.visitChildren(ctx)

```

For example:

Test	Result
"Let a: Int;"	Program([VarDecl(Id(a), IntType)])

Answer: (penalty regime: 0, 5, 10, 15, 20, ... %)

```
15 # Visit a parse tree produced by CSELParse#decl.
16
17 def visitDecl(self, ctx:CSELParse.DeclContext):
18
19     return self.visitChildren(ctx)
20
21 # Visit a parse tree produced by CSELParse#decltail.
22
23 def visitDecltail(self, ctx:CSELParse.DecltailContext):
24
25     return self.visitChildren(ctx)
26
27 # Visit a parse tree produced by CSELParse#vardecl.
28
29 def visitVardecl(self, ctx:CSELParse.VardeclContext):
30
31     return self.visitChildren(ctx)
32
33 # Visit a parse tree produced by CSELParse#single_vardecls.
34
35 def visitSingle_vardecls(self, ctx:CSELParse.Single_vardeclsContext):
36
37     return self.visitChildren(ctx)
38
39 # Visit a parse tree produced by CSELParse#single_vardecl.
40
41 def visitSingle_vardecl(self, ctx:CSELParse.Single_vardeclContext):
42
43     return self.visitChildren(ctx)
44
45 # Visit a parse tree produced by CSELParse#single_vardecltail.
46
47 def visitSingle_vardecltail(self, ctx:CSELParse.Single_vardecltailContext):
48
49     return self.visitChildren(ctx)
50
51 # Visit a parse tree produced by CSELParse#constdecl.
52
53 def visitConstdecl(self, ctx:CSELParse.ConstdeclContext):
54
55     return self.visitChildren(ctx)
56
```

```
57 # Visit a parse tree produced by CSELParse#single_constdecl.
58
59 def visitSingle_constdecl(self, ctx:CSELParse.Single_constdeclContext):
60
61     return self.visitChildren(ctx)
62
63 # Visit a parse tree produced by CSELParse#expr.
64
65 def visitExpr(self, ctx:CSELParse.ExprContext):
66
67     return self.visitChildren(ctx)
68
69 # Visit a parse tree produced by CSELParse#funcdecl.
70
71 def visitFuncdecl(self, ctx:CSELParse.FuncdeclContext):
72
73     return self.visitChildren(ctx)
74
75 # Visit a parse tree produced by CSELParse#paramlist.
76
77 def visitParamlist(self, ctx:CSELParse.ParamlistContext):
78
79     return self.visitChildren(ctx)
```

Kiểm tra

Copyright 2007-2021 Trường Đại Học Bách Khoa - ĐHQG Tp.HCM. All Rights Reserved.

Địa chỉ: Nhà A1- 268 Lý Thường Kiệt, Phường 14, Quận 10, Tp.HCM.

Email: elarning@hcmut.edu.vn

Phát triển dựa trên hệ thống Moodle