



BÁCH KHOA E-LEARNING

[Trang của tôi](#) / [Khóa học](#) / [Học kỳ I năm học 2021-2022 \(Semester 1 - Academic year 2021-2022\)](#)

/ [Chương Trình Chất Lượng Cao dạy bằng Tiếng Anh \(High-Quality training program\)](#)

/ [Khoa Khoa học và Kỹ thuật Máy tính \(Faculty of Computer Science and Engineering.\)](#) / [Khoa Học Máy Tính](#)

/ [Principles of Programming Languages \(CO3005\) _Nguyễn Hứa Phùng_\(CC_HK211\)](#) / 6-AST / [AST Programming](#)

Đã bắt đầu vào lúc	Tuesday, 21 September 2021, 1:46 PM
Tình trạng	Đã hoàn thành
Hoàn thành vào lúc	Tuesday, 21 September 2021, 4:25 PM
Thời gian thực hiện	2 giờ 38 phút
Điểm	6,00/6,00
Điểm	10,00 của 10,00 (100%)

Câu hỏi 1

Chính xác

Điểm 1,00 của 1,00

Given the grammar of MP as follows:

program: vardecls EOF;

vardecls: vardecl vardecltail;

vardecltail: vardecl vardecltail | ;

vardecl: mptype ids ';' ;

mptype: INTTYPE | FLOATTYPE;

ids: ID ',' ids | ID;

INTTYPE: 'int';

FLOATTYPE: 'float';

ID: [a-z]+ ;

Please copy the following class into your answer and modify the bodies of its methods to count the terminal nodes in the parse tree? Your code starts at line 10.

class TerminalCount(MPVisitor):

def visitProgram(self,ctx:MPParser.ProgramContext):

return None

def visitVardecls(self,ctx:MPParser.VardeclsContext):

return None

def visitVardecltail(self,ctx:MPParser.VardecltailContext):

return None

def visitVardecl(self,ctx:MPParser.VardeclContext):

return None

def visitMptype(self,ctx:MPParser.MptypeContext):

return None

def visitIds(self,ctx:MPParser.IdsContext):

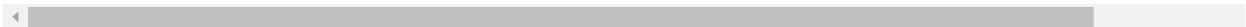
return None

Answer: (penalty regime: 0 %)

```

1 class TerminalCount(MPVisitor):
2     def visitProgram(self,ctx:MPParser.ProgramContext):
3         return self.visit(ctx.vardecls())
4     def visitVardecls(self,ctx:MPParser.VardeclsContext):
5         return self.visit(ctx.vardecl()) + self.visit(ctx.vardecltail())
6     def visitVardecltail(self,ctx:MPParser.VardecltailContext):
7         if ctx.getChildCount() == 0:
8             return 1
9         return (self.visit(ctx.vardecl()) if ctx.vardecl() else 0) + (self.visit(ctx.vardecltail()) if ctx.v
10    def visitVardecl(self,ctx:MPParser.VardeclContext):
11        return 1 + self.visit(ctx.mptype()) + self.visit(ctx.ids())
12    def visitMptype(self,ctx:MPParser.MptypeContext):
13        return 1
14    def visitIds(self,ctx:MPParser.IdsContext):
15        if ctx.getChildCount() == 1:
16            return 1
17        return 2 + (self.visit(ctx.ids()) if ctx.ids() else 0)

```



	Test	Expected	Got	
✓	"int a;"	4	4	✓
✓	""int a,b;""	6	6	✓
✓	"int a;float b;"	7	7	✓
✓	"int a,b;float c;"	9	9	✓
✓	"int a,b;float c,d,e;"	13	13	✓

Passed all tests! ✓

Chính xác

Điểm cho bài nộp này: 1,00/1,00.

Câu hỏi 2

Chính xác

Điểm 1,00 của 1,00

Given the grammar of MP as follows:

program: vardecls EOF;

vardecls: vardecl vardecltail;

vardecltail: vardecl vardecltail | ;

vardecl: mptype ids ';' ;

mptype: INTTYPE | FLOATTYPE;

ids: ID ',' ids | ID;

INTTYPE: 'int';

FLOATTYPE: 'float';

ID: [a-z]+ ;

Please copy the following class into your answer and modify the bodies of its methods to return the height of the parse tree? Your code starts at line 10.

```
class TerminalCount(MPVisitor):

    def visitProgram(self,ctx:MPParser.ProgramContext):

        return None

    def visitVardecls(self,ctx:MPParser.VardeclsContext):

        return None

    def visitVardecltail(self,ctx:MPParser.VardecltailContext):

        return None

    def visitVardecl(self,ctx:MPParser.VardeclContext):

        return None

    def visitMptype(self,ctx:MPParser.MptypeContext):

        return None

    def visitIds(self,ctx:MPParser.IdsContext):

        return None
```

Answer: (penalty regime: 0 %)

```
1 class TerminalCount(MPVisitor):
2     def visitProgram(self,ctx:MPParser.ProgramContext):
3         return self.visit(ctx.vardecls())
4     def visitVardecls(self,ctx:MPParser.VardeclsContext):
5         _1 = self.visit(ctx.vardecl())
6         _2 = self.visit(ctx.vardecltail())
7         return 1 + (_1 if _1 >= _2 else _2)
8     def visitVardecltail(self,ctx:MPParser.VardecltailContext):
9         if ctx.getChildCount() == 0:
10            return 1
11         _1 = self.visit(ctx.vardecl())
12         _2 = self.visit(ctx.vardecltail())
13         return 1 + (_1 if _1 >= _2 else _2)
14     def visitVardecl(self,ctx:MPParser.VardeclContext):
15         _1 = self.visit(ctx.mptype())
16         _2 = self.visit(ctx.ids())
17         return 1 + (_1 if _1 >= _2 else _2)
18     def visitMptype(self,ctx:MPParser.MptypeContext):
19         return 2
20     def visitIds(self,ctx:MPParser.IdsContext):
21         if ctx.getChildCount() == 1:
```

```
22 |         return 2
23 |     return 1 + self.visit(ctx.ids())
```

	Test	Expected	Got	
✓	"int a;"	4	4	✓
✓	""int a,b;""	5	5	✓
✓	"int a;float b;"	5	5	✓
✓	"int a,b;float c;"	5	5	✓
✓	"int a,b;float c,d,e;"	7	7	✓

Passed all tests! ✓

Chính xác

Điểm cho bài nộp này: 1,00/1,00.

Câu hỏi 3

Chính xác

Điểm 1,00 của 1,00

Given the grammar of MP as follows:

program: exp EOF;

exp: term ASSIGN exp | term;

term: factor COMPARE factor | factor;

factor: factor ANDOR operand | operand;

operand: ID | INTLIT | BOOLIT | '(' exp ')';

INTLIT: [0-9]+ ;

BOOLIT: 'True' | 'False' ;

ANDOR: 'and' | 'or' ;

ASSIGN: '+=' | '-=' | '&=' | '|=' | ':=' ;

COMPARE: '=' | '<>' | '>=' | '<=' | '<' | '>' ;

ID: [a-z]+ ;

and AST classes as follows:

class Expr(ABC):

class Binary(Expr): #op:string;left:Expr;right:Expr

class Id(Expr): #value:string

class IntLiteral(Expr): #value:int

class BooleanLiteral(Expr): #value:boolean

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

class ASTGeneration(MPVisitor):

def visitProgram(self,ctx:MPParser.ProgramContext):

return None

def visitExp(self,ctx:MPParser.ExpContext):

return None

def visitTerm(self,ctx:MPParser.TermContext):

return None

def visitFactor(self,ctx:MPParser.FactorContext):

return None

def visitOperand(self,ctx:MPParser.OperandContext):

return None

Answer: (penalty regime: 0 %)

```

1 class ASTGeneration(MPVisitor):
2
3     def visitProgram(self,ctx:MPParser.ProgramContext):
4
5         return self.visit(ctx.exp())
6
7     def visitExp(self,ctx:MPParser.ExpContext):
8         if ctx.getChildCount() == 1:
9             return self.visit(ctx.term())
10
11         return Binary(ctx.ASSIGN().getText(),self.visit(ctx.term()),self.visit(ctx.exp()))
12

```

```
13 def visitTerm(self,ctx:MPParser.TermContext):
14     if ctx.getChildCount() == 1:
15         return self.visit(ctx.factor()[0])
16     return Binary(ctx.COMPARE().getText(),self.visit(ctx.factor()[0]),self.visit(ctx.factor()[1]))
17
18 def visitFactor(self,ctx:MPParser.FactorContext):
19     if ctx.getChildCount() == 1:
20         return self.visit(ctx.operand())
21     return Binary(ctx.ANDOR().getText(),self.visit(ctx.factor()),self.visit(ctx.operand()))
22
23 def visitOperand(self,ctx:MPParser.OperandContext):
24     if ctx.exp():
```

	Test	Expected	Got
✓	"a := b := 4"	Binary(:=,Id(a),Binary(:=,Id(b),IntLiteral(4)))	Binary(:=,Id(a),Binary(:=,Id(b),IntLiteral(4)))
✓	""a += b -= a and (b > 3)""	Binary(+,Id(a),Binary(- =,Id(b),Binary(and,Id(a),Binary(>,Id(b),IntLiteral(3)))))	Binary(+,Id(a),Binary(- =,Id(b),Binary(and,Id(a),Binary(>,Id(b),IntLiteral(3)))))
✓	"a or b and True"	Binary(and,Binary(or,Id(a),Id(b)),BooleanLiteral(True))	Binary(and,Binary(or,Id(a),Id(b)),BooleanLiteral(True))

Passed all tests! ✓

Chính xác

Điểm cho bài nộp này: 1,00/1,00.

Câu hỏi 4

Chính xác

Điểm 1,00 của 1,00

Given the grammar of MP as follows:

program: vardecls EOF;

vardecls: vardecl vardecltail;

vardecltail: vardecl vardecltail | ;

vardecl: mptype ids ';' ;

mptype: INTTYPE | FLOATTYPE;

ids: ID ',' ids | ID;

INTTYPE: 'int';

FLOATTYPE: 'float';

ID: [a-z]+ ;

and AST classes as follows:

class Program: #decl: list(VarDecl)

class Type(ABC): pass

class IntType(Type): pass

class FloatType(Type): pass

class VarDecl: #variable: Id; varType: Type

class Id: #name: str

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

class ASTGeneration(MPVisitor):

def visitProgram(self, ctx: MPParser.ProgramContext):

return None

def visitVardecls(self, ctx: MPParser.VardeclsContext):

return None

def visitVardecltail(self, ctx: MPParser.VardecltailContext):

return None

def visitVardecl(self, ctx: MPParser.VardeclContext):

return None

def visitMptype(self, ctx: MPParser.MptypeContext):

return None

def visitIds(self, ctx: MPParser.IdsContext):

return None

Answer: (penalty regime: 0 %)

```

1 class ASTGeneration(MPVisitor):
2
3     def visitProgram(self, ctx: MPParser.ProgramContext):
4
5         return Program(self.visit(ctx.vardecls()))
6
7     def visitVardecls(self, ctx: MPParser.VardeclsContext):
8
9         return self.visit(ctx.vardecl()) + (self.visit(ctx.vardecltail()) if ctx.vardecltail() else [])
10

```



```
11 def visitVardecltail(self,ctx:MPParser.VardecltailContext):
12
13     return (self.visit(ctx.vardecl()) if ctx.vardecl() else []) + \
14            (self.visit(ctx.vardecltail()) if ctx.vardecltail() else [])
15
16 def visitVardecl(self,ctx:MPParser.VardeclContext):
17     return [VarDecl(_id,self.visit(ctx.mptype())) for _id in self.visit(ctx.ids())]
18
19 def visitMptype(self,ctx:MPParser.MptypeContext):
20     if ctx.INTTYPE():
21         return IntType()
22     if ctx.FLOATTYPE():
23         return FloatType()
24
```

	Test	Expected
✓	"int a;"	Program([VarDecl(Id(a),IntType)])
✓	"""int a,b;"""	Program([VarDecl(Id(a),IntType),VarDecl(Id(b),IntType)])
✓	"int a;float b;"	Program([VarDecl(Id(a),IntType),VarDecl(Id(b),FloatType)])
✓	"int a,b;float c;"	Program([VarDecl(Id(a),IntType),VarDecl(Id(b),IntType),VarDecl(Id(c),FloatType)])
✓	"int a,b;float c,d,e;"	Program([VarDecl(Id(a),IntType),VarDecl(Id(b),IntType),VarDecl(Id(c),FloatType),VarDecl(Id(d),FloatType),VarDecl(

Passed all tests! ✓

Chính xác

Điểm cho bài nộp này: 1,00/1,00.

Câu hỏi 5

Chính xác

Điểm 1,00 của 1,00

Given the grammar of MP as follows:

program: exp EOF;

exp: (term ASSIGN)* term;

term: factor COMPARE factor | factor;

factor: operand (ANDOR operand)*;

operand: ID | INTLIT | BOOLIT | '(' exp ')';

INTLIT: [0-9]+ ;

BOOLIT: 'True' | 'False' ;

ANDOR: 'and' | 'or' ;

ASSIGN: '+=' | '-=' | '&=' | '|=' | ':=' ;

COMPARE: '=' | '<>' | '>=' | '<=' | '<' | '>' ;

ID: [a-z]+ ;

and AST classes as follows:

class Expr(ABC):

class Binary(Expr): #op:string;left:Expr;right:Expr

class Id(Expr): #value:string

class IntLiteral(Expr): #value:int

class BooleanLiteral(Expr): #value:boolean

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

class ASTGeneration(MPVisitor):

def visitProgram(self,ctx:MPParser.ProgramContext):

return None

def visitExp(self,ctx:MPParser.ExpContext):

return None

def visitTerm(self,ctx:MPParser.TermContext):

return None

def visitFactor(self,ctx:MPParser.FactorContext):

return None

def visitOperand(self,ctx:MPParser.OperandContext):

return None

Answer: (penalty regime: 0 %)

```

11         binary(ele[0].getText(),
12               ele[1].accept(self),
13               acc),
14         zip (ctx.ASSIGN()[::-1], list(ctx.term()[0:-1])[::-1]),
15         ctx.term()[-1].accept(self))
16
17     def visitTerm(self,ctx:MPParser.TermContext):
18         if ctx.getChildCount() == 1:
19             return self.visit(ctx.factor()[0])
20         return Binary(ctx.COMPARE().getText(),self.visit(ctx.factor()[0]),self.visit(ctx.factor()[1]))
21
22     def visitFactor(self,ctx:MPParser.FactorContext):
23         return reduce(lambda acc,ele:

```

```
23         return result(lambda acc, ele:
24             Binary(ele[0].getText(),
25                 acc,
26                 ele[1].accept(self)),
27                 zip (ctx.ANDOR(), ctx.operand()[1:]),
28                 ctx.operand(0).accept(self))
29
30 def visitOperand(self, ctx:MPParser.OperandContext):
31     if ctx.exp():
32         return self.visit(ctx.exp())
33     if ctx.ID():
34         return Id(ctx.ID().getText())
35     if ctx.INT_LITERAL():
```

	Test	Expected	Got
✓	"a := b := 4"	Binary(:=,Id(a),Binary(:=,Id(b),IntLiteral(4)))	Binary(:=,Id(a),Binary(:=,Id(b),IntLiteral(4)))
✓	"""a += b -= a and (b > 3)"""	Binary(+,Id(a),Binary(- =,Id(b),Binary(and,Id(a),Binary(>,Id(b),IntLiteral(3)))))	Binary(+,Id(a),Binary(- =,Id(b),Binary(and,Id(a),Binary(>,Id(b),IntLiteral(3)))))
✓	"a or b and True"	Binary(and,Binary(or,Id(a),Id(b)),BooleanLiteral(True))	Binary(and,Binary(or,Id(a),Id(b)),BooleanLiteral(True))

Passed all tests! ✓

Chính xác

Điểm cho bài nộp này: 1,00/1,00.

Câu hỏi 6

Chính xác

Điểm 1,00 của 1,00

Given the grammar of MP as follows:

program: vardecl+ EOF;

vardecl: mptype ids ';' ;

mptype: INTTYPE | FLOATTYPE;

ids: ID (';' ID)*;

INTTYPE: 'int';

FLOATTYPE: 'float';

ID: [a-z]+ ;

and AST classes as follows:

class Program: #decl: list(VarDecl)

class Type(ABC): pass

class IntType(Type): pass

class FloatType(Type): pass

class VarDecl: #variable: Id; varType: Type

class Id: #name: str

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

```
class ASTGeneration(MPVisitor):
    def visitProgram(self, ctx: MPParser.ProgramContext):
        return None

    def visitVardecl(self, ctx: MPParser.VardeclContext):
        return None

    def visitMptype(self, ctx: MPParser.MptypeContext):
        return None

    def visitIds(self, ctx: MPParser.IdsContext):
        return None
```

Answer: (penalty regime: 0 %)

```
1 def flatten(lst):
2     if not isinstance(lst, list):
3         return [lst]
4     if len(lst)==0:
5         return []
6     if len(lst)==1:
7         return flatten(lst[0])
8
9     return flatten(lst[0]) + flatten(lst[1:])
10
11 class ASTGeneration(MPVisitor):
12
13     def visitProgram(self, ctx: MPParser.ProgramContext):
14
15         return Program(flatten([self.visit(x) for x in ctx.vardecl()]))
16
17     def visitVardecl(self, ctx: MPParser.VardeclContext):
18         return [VarDecl(_id, self.visit(ctx.mptype())) for _id in self.visit(ctx.ids())]
19
20     def visitMptype(self, ctx: MPParser.MptypeContext):
21         if ctx.INTTYPE():
22             return IntType()
```

```

22 |         return IntType();
23 |     if ctx.FLOATTYPE():
24 |         return FloatType()
25 |

```

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: elearning@hcmut.edu.vn

Phát triển dựa trên hệ thống Moodle

✓	"int a;float b;"	Program([VarDecl(Id(a),IntType),VarDecl(Id(b),FloatType)])
✓	"int a,b;float c;"	Program([VarDecl(Id(a),IntType),VarDecl(Id(b),IntType),VarDecl(Id(c),FloatType)])
✓	"int a,b;float c,d,e;"	Program([VarDecl(Id(a),IntType),VarDecl(Id(b),IntType),VarDecl(Id(c),FloatType),VarDecl(Id(d),FloatType),VarDecl(

Passed all tests! ✓

Chính xác

Điểm cho bài nộp này: 1,00/1,00.

◀ AST Quiz

Chuyển tới...

[Link video of session 21/09/2021 ▶](#)