BÁCH KHOA E-LEARNING

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 %)

```
class TerminalCount(MPVisitor):
    def visitProgram(self,ctx:MPParser.ProgramContext):
        return self.visit(ctx.vardecls())
    def visitVardecls(self,ctx:MPParser.VardeclsContext):
        return self.visit(ctx.vardecl()) + self.visit(ctx.vardecltail())
    def visitVardecltail(self,ctx:MPParser.VardecltailContext):
        if ctx.getChildCount() == 0:
            return 1
        return (self.visit(ctx.vardecl()) if ctx.vardecl() else 0) + (self.visit(ctx.vardecltail()) if ctx.v
    def visitVardecl(self,ctx:MPParser.VardeclContext):
        return 1 + self.visit(ctx.mptype()) + self.visit(ctx.ids())
    def visitMptype(self,ctx:MPParser.MptypeContext):
        return 1
    def visitIds(self,ctx:MPParser.IdsContext):
        if ctx.getChildCount() == 1:
            return 1
        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 %)

```
class TerminalCount(MPVisitor):
    def visitProgram(self,ctx:MPParser.ProgramContext):
        return self.visit(ctx.vardecls())
    def visitVardecls(self,ctx:MPParser.VardeclsContext):
        _1 = self.visit(ctx.vardecl())
        _2 = self.visit(ctx.vardecltail())
        return 1 + (_1 if _1 >= _2 else _2)
    def visitVardecltail(self,ctx:MPParser.VardecltailContext):
        if ctx.getChildCount() == 0:
            return 1
        _1 = self.visit(ctx.vardecl())
        _2 = self.visit(ctx.vardecltail())
        return 1 + (_1 if _1 >= _2 else _2)
    def visitVardecl(self,ctx:MPParser.VardeclContext):
        _1 = self.visit(ctx.mptype())
        _2 = self.visit(ctx.ids())
        return 1 + (_1 if _1 >= _2 else _2)
    def visitMptype(self,ctx:MPParser.MptypeContext):
        return 2
    def visitIds(self,ctx:MPParser.IdsContext):
        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 %)

```
class ASTGeneration(MPVisitor):

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

        return self.visit(ctx.exp())

    def visitExp(self,ctx:MPParser.ExpContext):
        if ctx.getChildCount() == 1:
            return self.visit(ctx.term())

        return Binary(ctx.ASSIGN().getText(),self.visit(ctx.term()),self.visit(ctx.exp()))

```

```
    def visitTerm(self,ctx:MPParser.TermContext):
        if ctx.getChildCount() == 1:
            return self.visit(ctx.factor()[0])
        return Binary(ctx.COMPARE().getText(),self.visit(ctx.factor()[0]),self.visit(ctx.factor()[1]))

    def visitFactor(self,ctx:MPParser.FactorContext):
        if ctx.getChildCount() == 1:
            return self.visit(ctx.operand())
        return Binary(ctx.ANDOR().getText(),self.visit(ctx.factor()),self.visit(ctx.operand()))

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

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

        return (self.visit(ctx.vardecl()) if ctx.vardecl() else []) + \
            (self.visit(ctx.vardecltail()) if ctx.vardecltail() else [])

    def visitVardecl(self,ctx:MPParser.VardeclContext):
        return [VarDecl(_id,self.visit(ctx.mptype())) for _id in self.visit(ctx.ids())]

    def visitMptype(self,ctx:MPParser.MptypeContext):
        if ctx.INTTYPE():
            return IntType()
        if ctx.FLOATTYPE():
            return FloatType()

```

	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 %)

```
            Binary(ele[0].getText(),
            ele[1].accept(self),
            acc),
            zip (ctx.ASSIGN()[::-1], list(ctx.term()[0:-1])[::-1]),
            ctx.term()[-1].accept(self))

    def visitTerm(self,ctx:MPParser.TermContext):
        if ctx.getChildCount() == 1:
            return self.visit(ctx.factor()[0])
        return Binary(ctx.COMPARE().getText(),self.visit(ctx.factor()[0]),self.visit(ctx.factor()[1]))

    def visitFactor(self,ctx:MPParser.FactorContext):
        return reduce(lambda acc ,ele:
```

```
            Binary(ele[0].getText(),
                acc,
                ele[1].accept(self)),
                zip (ctx.ANDOR(), ctx.operand()[1:]),
                ctx.operand(0).accept(self))

    def visitOperand(self,ctx:MPParser.OperandContext):
        if ctx.exp():
            return self.visit(ctx.exp())
        if ctx.ID():
            return Id(ctx.ID().getText())
```

	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 %)

```
def flatten(lst):
    if not isinstance(lst,list):
        return [lst]
    if len(lst)==0:
        return []
    if len(lst)==1:
        return flatten(lst[0])

    return flatten(lst[0]) + flatten(lst[1:])

class ASTGeneration(MPVisitor):

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

        return Program(flatten([self.visit(x) for x in ctx.vardecl()]))

    def visitVardecl(self,ctx:MPParser.VardeclContext):
        return [VarDecl(_id,self.visit(ctx.mptype())) for _id in self.visit(ctx.ids())]

    def visitMptype(self,ctx:MPParser.MptypeContext):
        if ctx.INTTYPE():
            return IntType()
```

```
        if ctx.FLOATTYPE():
            return FloatType()
```


✔	"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 ►