

[🏠 Trang chủ](#)[Trang của tôi](#) » [Học kỳ I năm học 2020-2021 \(Semester 1 - Academic year 2020-2021\)](#) »[Đại Học Chính Quy \(Bachelor program \(Full-time study\)\)](#) »[Khoa Khoa học và Kỹ thuật Máy tính \(Faculty of Computer Science and Engineering\)](#) »[Nguyên lý ngôn ngữ lập trình \(CO3005\)\\_Trần Ngọc Bảo Duy \(DH\\_HK201\)](#) » [Tên, Tầm vực và Môi trường tham khảo \(Tuần 8\)](#) »

Programming Code: Name

**Đã bắt đầu vào lúc** Monday, 23 November 2020, 2:55 PM**Tình trạng** Đã hoàn thành**Hoàn thành vào lúc** Monday, 23 November 2020, 3:12 PM**Thời gian thực hiện** 16 phút 23 giây**Điểm** 4,00/4,00**Điểm** **10,00** của 10,00 (**100%**)

**Câu hỏi 1**

Chính xác

Điểm 1,00 của 1,00

Let AST of a programming language be defined as follows:

```
class Program: #decl:List[Decl]
class Decl(ABC): #abstract class
class VarDecl(Decl): #name:str,typ:Type
class ConstDecl(Decl): #name:str,val:Lit
class Type(ABC): #abstract class
class IntType(Type)
class FloatType(Type)
class Lit(ABC): #abstract class
class IntLit(Lit): #val:int
```

and exception RedeclaredDeclaration:

```
class RedeclaredDeclaration(Exception): #name:str
```

Implement the methods of the following class Visitor to travel on the above ASST to detect redeclared declarations (throw exception RedeclaredDeclaration):

```
class StaticCheck(Visitor):
    def visitProgram(self,ctx:Program,o:object): pass
    def visitVarDecl(self,ctx:VarDecl,o:object):pass
    def visitConstDecl(self,ctx:ConstDecl,o:object):pass
    def visitIntType(self,ctx:IntType,o:object):pass
    def visitFloatType(self,ctx:FloatType,o:object):pass
    def visitIntLit(self,ctx:IntLit,o:object):pass
```

Your code starts at line 40

**For example:**

**Test**

```
x = Program([VarDecl("a",IntType()),ConstDecl("b",IntLit(3)),VarDecl("a",FloatTy
```

**Answer:** (penalty regime: 10, 20, ... %)

```
1 from functools import reduce
2 class StaticCheck(Visitor):
3
4     def visitProgram(self,ctx:Program,o:object):
5         reduce(lambda acc, ele: acc + [self.visit(ele, acc)], ctx.de
6
7     def visitVarDecl(self,ctx:VarDecl,o:object):
8         if list(filter(lambda x: x.name == ctx.name, o)):
9             raise RedeclaredDeclaration(ctx.name)
10        return ctx
11
12    def visitConstDecl(self,ctx:ConstDecl,o:object):
13        if list(filter(lambda x: x.name == ctx.name, o)):
```

```
14         raise RedeclaredDeclaration(ctx.name)
15     return ctx
```

	Test
✓	x = Program([VarDecl("a",IntType()),ConstDecl("b",IntLit(3)),VarDecl("a",F]

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

Let AST of a programming language be defined as follows:

```
class Program: #decl:List[Decl]
class Decl(ABC): #abstract class
class VarDecl(Decl): #name:str,typ:Type
class ConstDecl(Decl): #name:str,val:Lit
class Type(ABC): #abstract class
class IntType(Type)
class FloatType(Type)
class Lit(ABC): #abstract class
class IntLit(Lit): #val:int
```

and exceptions:

```
class RedeclaredVariable(Exception): #name:str
class RedeclaredConstant(Exception): #name:str
```

Implement the methods of the following class Visitor to travel on the above ASST to detect redeclared declarations (throw the exception corresponding to the second declaration with the same name):

```
class StaticCheck(Visitor):
    def visitProgram(self,ctx:Program,o:object): pass
    def visitVarDecl(self,ctx:VarDecl,o:object):pass
    def visitConstDecl(self,ctx:ConstDecl,o:object):pass
    def visitIntType(self,ctx:IntType,o:object):pass
    def visitFloatType(self,ctx:FloatType,o:object):pass
    def visitIntLit(self,ctx:IntLit,o:object):pass
```

Your code starts at line 45

**For example:**

**Test**

```
x = Program([VarDecl("a",IntType()),ConstDecl("b",IntLit(3)),VarDecl("a",FloatTy
```

**Answer:** (penalty regime: 10, 20, ... %)

```
1  from functools import reduce
2  class StaticCheck(Visitor):
3
4      def visitProgram(self,ctx:Program,o:object):
5          reduce(lambda acc, ele: acc + [self.visit(ele, acc)], ctx.de
6
7      def visitVarDecl(self,ctx:VarDecl,o:object):
8          if list(filter(lambda x: x.name == ctx.name, o)):
9              raise RedeclaredVariable(ctx.name)
10         return ctx
```

```
11     def visitConstDecl(self,ctx:ConstDecl,o:object):
12         if list(filter(lambda x: x.name == ctx.name, o)):
13             raise RedeclaredConstant(ctx.name)
14         return ctx
15
```

	Test
✓	x = Program([VarDecl("a",IntType()),ConstDecl("b",IntLit(3)),VarDecl("a",F]

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

Let AST of a programming language be defined as follows:

```
class Program: #decl:List[Decl]
class Decl(ABC): #abstract class
class VarDecl(Decl): #name:str,typ:Type
class ConstDecl(Decl): #name:str,val:Lit
class FuncDecl(Decl): #name:str,param:List[VarDecl],body:List[Decl]
class Type(ABC): #abstract class
class IntType(Type)
class FloatType(Type)
class Lit(ABC): #abstract class
class IntLit(Lit): #val:int
and exceptions:
```

```
class RedeclaredVariable(Exception): #name:str
class RedeclaredConstant(Exception): #name:str
class RedeclaredFunction(Exception): #name:str
```

Implement the methods of the following class Visitor to travel on the above AST to detect redeclared declarations (throw the exception corresponding to the second declaration with the same name) in the same scope:

```
class StaticCheck(Visitor):
    def visitProgram(self,ctx:Program,o:object): pass
    def visitVarDecl(self,ctx:VarDecl,o:object):pass
    def visitConstDecl(self,ctx:ConstDecl,o:object):pass
    def visitFuncDecl(self,ctx:FuncDecl,o:object):pass
    def visitIntType(self,ctx:IntType,o:object):pass
    def visitFloatType(self,ctx:FloatType,o:object):pass
    def visitIntLit(self,ctx:IntLit,o:object):pass
```

Your code starts at line 55

**For example:**

**Test**

```
x = Program([VarDecl("a",IntType()),ConstDecl("b",IntLit(3)),FuncDecl("a",[],[])])
```

**Answer:** (penalty regime: 10, 20, ... %)

```
1 from functools import reduce
2 class StaticCheck(Visitor):
3
4     def visitProgram(self,ctx:Program,o:object):
5         reduce(lambda acc, ele: acc + [self.visit(ele, acc)], ctx.de
6
```

```

7      def visitVarDecl(self,ctx:VarDecl,o:object):
8          if list(filter(lambda x: x.name == ctx.name, o)):
9              raise RedeclaredVariable(ctx.name)
10         return ctx
11
12     def visitConstDecl(self,ctx:ConstDecl,o:object):
13         if list(filter(lambda x: x.name == ctx.name, o)):
14             raise RedeclaredConstant(ctx.name)
15         return ctx
16     def visitFuncDecl(self,ctx:FuncDecl,o:object):
17         if list(filter(lambda x: x.name == ctx.name, o)):
18             raise RedeclaredFunction(ctx.name)
19         reduce(lambda acc, ele: acc + [self.visit(ele, acc)], ctx.pa
20         return ctx

```

	Test
✓	x = Program([VarDecl("a",IntType()),ConstDecl("b",IntLit(3)),FuncDecl("a",[

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

Let AST of a programming language be defined as follows:

```
class Program: #decl:List[Decl]
class Decl(ABC): #abstract class
class VarDecl(Decl): #name:str,typ:Type
class ConstDecl(Decl): #name:str,val:Lit
class FuncDecl(Decl): #name:str,param:List[VarDecl],body:Tuple(List[Decl],List[Expr])
class Type(ABC): #abstract class
class IntType(Type)
class FloatType(Type)
class Expr(ABC): #abstract class
class Lit(Expr): #abstract class
class IntLit(Lit): #val:int
class Id(Expr): #name:str
```

and exceptions:

```
class RedeclaredVariable(Exception): #name:str
class RedeclaredConstant(Exception): #name:str
class RedeclaredFunction(Exception): #name:str
class UndeclaredIdentifier(Exception): #name:str
```

Implement the methods of the following class Visitor to travel on the above AST to detect undeclared declarations (throw the exception UndeclaredIdentifier). Note that the redeclared declarations exception also is thrown if a redeclared declaration is detected:

```
class StaticCheck(Visitor):
    def visitProgram(self,ctx:Program,o:object): pass
    def visitVarDecl(self,ctx:VarDecl,o:object):pass
    def visitConstDecl(self,ctx:ConstDecl,o:object):pass
    def visitFuncDecl(self,ctx:FuncDecl,o:object):pass
    def visitIntType(self,ctx:IntType,o:object):pass
    def visitFloatType(self,ctx:FloatType,o:object):pass
    def visitIntLit(self,ctx:IntLit,o:object):pass
    def visitId(self,ctx:Id,o:object):pass
```

Your code starts at line 65

**For example:**

**Test**

```
x = Program([VarDecl("a",IntType()),ConstDecl("b",IntLit(3)),FuncDecl("a",[[]],([[]]
x = Program([VarDecl("b",IntType()),FuncDecl("a",[VarDecl("m",FloatType()),VarDe
```



**Answer:** (penalty regime: 10, 20, ... %)

```
1 from functools import reduce
2 class StaticCheck(Visitor):
3
4     def visitProgram(self,ctx:Program,o:object):
5         reduce(lambda acc, ele: [acc[0] + [self.visit(ele, acc)], ac
6
7     def visitVarDecl(self,ctx:VarDecl,o:object):
8         if list(filter(lambda x: x.name == ctx.name, o[0])):
9             raise RedeclaredVariable(ctx.name)
10        return ctx
11
12    def visitConstDecl(self,ctx:ConstDecl,o:object):
13        if list(filter(lambda x: x.name == ctx.name, o[0])):
14            raise RedeclaredConstant(ctx.name)
15        return ctx
16    def visitFuncDecl(self,ctx:FuncDecl,o:object):
17        if list(filter(lambda x: x.name == ctx.name, o[0])):
18            raise RedeclaredFunction(ctx.name)
19        o[1].append(ctx)
20
21
```

	Test
✓	x = Program([VarDecl("a",IntType()),ConstDecl("b",IntLit(3)),FuncDecl("a",[
✓	x = Program([VarDecl("b",IntType()),FuncDecl("a",[VarDecl("m",FloatType()),

Passed all tests! ✓

**Chính xác**

Điểm cho bài nộp này: 1,00/1,00.

**Copyright 2007-2014 BKĐT-Đại Học Bách Khoa Tp.HCM. All Rights Reserved.**

Địa chỉ: Nhà A1- 268 Lý Thường Kiệt, Phường 14, Quận 10, Tp.HCM. Email: [elearning@hcmut.edu.vn](mailto:elearning@hcmut.edu.vn)

Phát triển dựa trên hệ thống Moodle