| | |
|---|---|
| **Trạng thái** | Đã xong |
| **Bắt đầu vào lúc** | Thứ Năm, 17 tháng 4 2025, 7:36 AM |
| **Kết thúc lúc** | Thứ Năm, 17 tháng 4 2025, 8:44 AM |
| **Thời gian thực hiện** | 1 giờ 8 phút |
| **Điểm** | 17,00/18,00 |
| **Điểm** | **9,44** trên 10,00 (**94,44**%) |

Thông tin

Cho các lớp của AST được khai báo như sau:

```
class AST(ABC): pass

class Decl(AST): pass
class Type(AST): pass

class TypeDecl(Decl):  # name: str, rhs: Type
    def __init__(self, name, rhs): self.name = name; self.rhs = rhs
    def accept(self, v, o): return v.visitTypeDecl(self, o)

class VarDecl(Decl):  # name: str, typ: Type
    def __init__(self, name, typ): self.name = name; self.typ = typ
    def accept(self, v, o): return v.visitVarDecl(self, o)

class Block(Decl):  # ele: List[Decl]
    def __init__(self, ele): self.ele = ele
    def accept(self, v, o): return v.visitBlock(self, o)

class IntType(Type):
    def accept(self, v, o): return v.visitIntType(self, o)

class FloatType(Type):
    def accept(self, v, o): return v.visitFloatType(self, o)

class Id(Type):  # name: str
    def __init__(self, name): self.name = name
    def accept(self, v, o): return v.visitId(self, o)

class Visitor(ABC):
    def visit(self, ctx, o): return ctx.accept(self, o)

class UndeclaredType(Exception):
    def __init__(self, name): self.name = name
    def __str__(self): return "Undeclared type: " + str(self.name)

class Symbol:
    def __init__(self, name, typ): self.name = name; self.typ = typ
```

**Yêu cầu:** Viết lớp `StaticCheck` để kiểm tra xem tên kiểu có được khai báo hợp lệ hay không. Một tên kiểu chỉ hợp lệ nếu nó đã được khai báo bởi `TypeDecl` trong cùng `Block` hoặc một `Block` bao bên ngoài. Nếu không thỏa, ném lỗi `UndeclaredType`.

**Ví dụ:**

```
Block([
    VarDecl("a", IntType()),
    TypeDecl("vd", FloatType()),
    Block([
        VarDecl("b", FloatType()),
        VarDecl("c", Id("vd"))   # Hợp lệ
    ]),
    Block([
        VarDecl("d", IntType()),
        TypeDecl("vd1", IntType())
    ]),
    VarDecl("e", Id("vd"))      # Hợp lệ
    # Nếu dùng Id("vd1") ở đây thì sai
])
```

và lớp StaticChecker được hiện thực một phần như sau:

```
class StaticCheck(Visitor):
    def visitBlock(self, ast: Block, o):
        def reducer(acc, decl):
            res = self.visit(decl, acc)
            return acc if res is None else ___(1)___

        reduce(
            reducer,
            ast.ele,
            ___(2)___
        )

    def visitTypeDecl(self, ast: TypeDecl, o):
        return [___(3)___] + o[1:]

    def visitVarDecl(self, ast: VarDecl, o):
        if type(ast.typ) is ___(4)___:
            self.visit(ast.typ, o)
        return None

    def visitId(self, ast: Id, o):
        if next(
            filter(
                lambda sym: ast.name == sym.name,
                [sym for scope in o for sym in scope]
            ),
            ___(5)___
        ):
            return
        raise UndeclaredType(ast.name)
```

Câu hỏi **1**

Sai

Đạt điểm 0,00 trên 1,00

Trong đoạn mã `visitId`, mục đích của dòng lệnh sau là gì?

```
if next(filter(lambda sym: ast.name == sym.name, [sym for scope in o for sym in scope]), ___________):
```

Select one:

- a. Kiểm tra xem ast.name có được khai báo ở block hiện tại hay không
- b. Kiểm tra xem ast.name có phải là tên biến hợp lệ không ✕
- c. Tìm trong tất cả các kiểu đã khai báo và trả về kiểu phù hợp với ast.name
- d. Kiểm tra xem tên kiểu ast.name có được khai báo ở bất kỳ block nào bên ngoài hay không

Câu hỏi **2**

Đúng

Đạt điểm 1,00 trên 1,00

Trong `visitBlock`, tại sao lại truyền vào `[[]] + o` làm tham số `o` mới khi gọi `reduce`?

Select one:

- a. Để xóa tất cả các kiểu đã khai báo trước đó
- b. Để thêm một scope mới (một block mới) vào môi trường tham khảo ✓
- c. Để gom tất cả các scope lại thành một list duy nhất
- d. Để tạo ra một bản sao của o tránh sửa đổi trực tiếp

Câu hỏi **3**

Đúng

Đạt điểm 1,00 trên 1,00

Điền vào chỗ trống trong hàm `visitTypeDecl` để thêm kiểu mới vào block hiện tại:

```
def visitTypeDecl(self, ast: TypeDecl, o):
    return [__________] + o[1:]
```

Select one:

- a. o[1:] + [Symbol(ast.name, ast.rhs)]
- b. o[0] + [Symbol(ast.name, ast.rhs)] ✓
- c. [Symbol(ast.name, self.visit(ast.rhs))] + o[0]
- d. o + [Symbol(ast.name, ast.rhs)]

Câu hỏi **4**

Đúng

Đạt điểm 1,00 trên 1,00

Trong trường hợp khai báo `VarDecl("e", Id("vd1"))` xuất hiện sau `TypeDecl("vd1", IntType())` ở một block con trước đó, thì điều gì sẽ xảy ra?

Select one:

- a. Có lỗi vì vd1 không thuộc cùng block hoặc block cha ✓
- b. Không có lỗi vì vd1 đã được khai báo
- c. Không có lỗi vì tên không quan trọng, chỉ cần đúng kiểu
- d. Có lỗi vì vd1 không phải là kiểu hợp lệ

Câu hỏi **5**

Đúng

Đạt điểm 1,00 trên 1,00

Trong `visitVarDecl`, việc gọi `self.visit(ast.typ, o)` là cần thiết khi:

```
if type(ast.typ) is ________:
```

Select one:

- a. Symbol
- b. IntType
- c. Id ✓
- d. FloatType

Câu hỏi **6**

Đúng

Đạt điểm 1,00 trên 1,00

Điền vào chỗ trống để đảm bảo môi trường mới được khởi tạo chính xác khi bắt đầu xử lý một `Block`:

```
reduce(
    reducer,
    ast.ele,
    ________
)
```

Select one:

- a. [[]] + o
- b. [[]] if o is None else [[]] + o ✓
- c. [o]
- d. o + [[]]

Câu hỏi **7**

Đúng

Đạt điểm 1,00 trên 1,00

Hoàn chỉnh phần kiểm tra trong `visitId` để đảm bảo chỉ ném lỗi khi tên không có trong bất kỳ scope nào:

```
if next(
    filter(lambda sym: ast.name == sym.name, [sym for scope in o for sym in scope]),
    ________
):
```

Select one:

- a. []
- b. 0
- c. False
- d. None ✓

Câu hỏi **8**

Đúng

Đạt điểm 1,00 trên 1,00

Trong `visitBlock`, mỗi lần `visit(decl, acc)` trả về kết quả khác `None`, ta gán lại `acc` bằng:

```
def reducer(acc, decl):
    res = self.visit(decl, acc)
    return acc if res is None else ________
```

Select one:

- a. decl
- b. res ✓
- c. []
- d. o

Thông tin

Cho các lớp AST được khai báo như sau:

```
class AST(ABC): pass

class Program(AST):      # decls: List[Block]
class Decl(AST): pass

class Block(Decl):       # local: List[VarDecl], body: List[Assign]
class VarDecl(Decl):     # name: Id

class Exp(AST): pass

class Assign(Exp):       # lhs: Id, rhs: Exp
class Id(Exp):           # name: str
class NumLit(Exp):       # value: int or float
class BoolLit(Exp):      # value: bool
```

Một chương trình bao gồm nhiều `Block`. Trong mỗi `Block`, các biến được khai báo trong `local`, và các phát biểu gán (assignments) được liệt kê trong `body`.

**Yêu cầu:**

> Hãy viết lớp `StaticChecker` kế thừa `Visitor`, kiểm tra chương trình và ném lỗi `UnassignedVariable(Id(x))` nếu phát hiện có biến `a` đứng trước `b` trong danh sách khai báo nhưng `b` lại được gán giá trị trước `a` trong thân `Block`.

**Ví dụ:**

```
Program([
    Block(
        [VarDecl("a"), VarDecl("b"), VarDecl("c")],
        [Assign(Id("b"), Assign(Id("c"), NumLit(3)))]
    )
])
```

→ Ném lỗi: `Unassigned variable: Id(b)`

Và một đoạn bài làm bị khuyết một số chỗ:

```
class Symbol:
    def __init__(self, name, assigned):
        self.name = name
        self.assigned = assigned


class Access:
    def __init__(self, sym, is_left):
        self.sym = sym
        self.is_left = is_left


class StaticCheck(Visitor):
    def visitProgram(self, ctx: Program, o):
        for block in ctx.decls:
            self.visit(block, None)

    def visitBlock(self, ctx: Block, o):
        reduce(
            lambda acc, cur: self.visit(cur, Access(acc, False)),
            ctx.body,
            ____________  # [BLANK 1]
        )

    def visitVarDecl(self, ctx: VarDecl, o):
        return Symbol(ctx.name, False)

    def visitAssign(self, ctx: Assign, o):
        access = self.visit(ctx.rhs, Access(o.sym, False))
        return self.visit(ctx.lhs, Access(access.sym, True))

    def visitId(self, ctx: Id, o):
        if ____________:  # [BLANK 2]
            return o
        idx1, _ = next(filter(lambda x: ____________, enumerate(o.sym)), None)  # [BLANK 3]

        _, unassigned_sym = list(
            filter(lambda x: ____________, enumerate(o.sym))  # [BLANK 4]
        )[-1]

        if unassigned_sym is not None:
            raise UnassignedVariable(Id(unassigned_sym.name))

        return Access(
            reduce(
                lambda acc, cur: acc
                + [Symbol(ctx.name, ____________) if cur.name == ctx.name else cur],  # [BLANK 5]
                o.sym,
                [],
            ),
            o.is_left,
        )

    def visitNumLit(self, ctx: NumLit, o):
        return o

    def visitBoolLit(self, ctx: BoolLit, o):
        return o
```

Câu hỏi **9**

Đúng

Đạt điểm 1,00 trên 1,00

Trong lớp `StaticCheck`, vai trò chính của hàm `visitBlock` là gì?

Select one:

- a. Xây dựng danh sách các phép gán trong block
- b. Gán giá trị mặc định cho các biến khai báo
- c. Kiểm tra lỗi cú pháp trong chương trình
- d. Duyệt các biến khai báo và các phép gán, cập nhật trạng thái gán của biến ✓

Câu hỏi **10**

Đúng

Đạt điểm 1,00 trên 1,00

Tại sao cần sử dụng lớp `Symbol` trong bài giải này?

Select one:

- a. Để ánh xạ biến với tên hàm
- b. Để chứa giá trị số của biến
- c. Để theo dõi kiểu dữ liệu của biến
- d. Để theo dõi trạng thái đã gán hay chưa của biến ✓

Câu hỏi **11**

Đúng

Đạt điểm 1,00 trên 1,00

Biến `is_left` trong lớp `Access` có tác dụng gì?

Select one:

- a. Đánh dấu biến là biến toàn cục hay cục bộ
- b. Để đánh dấu lỗi nếu biến được sử dụng sau khi bị gán
- c. Xác định xem biến có nằm trong chương trình chính không
- d. Phân biệt giữa vế trái và vế phải của phép gán ✓

Câu hỏi **12**

Đúng

Đạt điểm 1,00 trên 1,00

Câu lệnh `raise UnassignedVariable(Id(unassigned_sym.name))` có vai trò gì?

Select one:

- a. Ném lỗi nếu một biến được gán trước một biến khai báo trước đó mà chưa được gán ✓
- b. Cảnh báo khi biến được sử dụng sai kiểu
- c. Xác nhận chương trình chạy đúng
- d. Dừng chương trình khi gặp lỗi cú pháp

Câu hỏi **13**

Đúng

Đạt điểm 1,00 trên 1,00

Kết quả của đoạn AST sau sẽ là gì?

```
Program([
    Block(
        [VarDecl("a"), VarDecl("b")],
        [Assign(Id("b"), Id("a"))]
    )
])
```

Select one:

- a. Lỗi: Unassigned variable: Id(b)
- b. Không có lỗi xảy ra
- c. Lỗi: Id(a) không khai báo
- d. Lỗi: Unassigned variable: Id(a) ✓

Câu hỏi **14**

Đúng

Đạt điểm 1,00 trên 1,00

Trong hàm `visitBlock`, giá trị khởi đầu (initializer) của hàm `reduce` là gì? (Đoạn mã cho [BLANK 1])

Select one:

- a. ctx.body
- b. o.sym
- c. [self.visit(x, None) for x in ctx.local] ✓
- d. []

Câu hỏi **15**

Đúng

Đạt điểm 1,00 trên 1,00

Điều kiện nào sau đây đảm bảo rằng biểu thức `visitId` chỉ xử lý ở vế trái phép gán? (Đoạn mã cho [BLANK 2])

Select one:

- a. if not o.is_left: ✓
- b. if o.sym is None:
- c. if o.is_left = False:
- d. if o.is_left:

Câu hỏi **16**

Đúng

Đạt điểm 1,00 trên 1,00

Điều kiện nào được dùng để tìm vị trí của biến đang xử lý trong bảng ký hiệu? (Đoạn mã tại [BLANK 3])

Select one:

- a. x[1].name == ctx.name ✓
- b. x[1].name != ctx.name
- c. x[0] < ctx.name
- d. x[0] == ctx.name

Câu hỏi **17**

Đúng

Đạt điểm 1,00 trên 1,00

Điều kiện nào sau đây dùng để tìm ra các biến **khai báo trước nhưng chưa được gán**? (Đoạn mã tại [BLANK 4])

Select one:

- a. x[0] > idx1 and x[1].assigned == False
- b. x[0] < idx1 or x[1].assigned== False
- c. x[0] < idx1 and x[1].assigned== False ✓
- d. x[1].assigned == True

Câu hỏi **18**

Đúng

Đạt điểm 1,00 trên 1,00

Trong biểu thức cập nhật bảng ký hiệu, giá trị `assigned` của biến được cập nhật thành gì? (Đoạn mã tại [BLANK 5])

Select one:

- a. False
- b. Symbol
- c. True ✓
- d. None