Nguyên lý ngôn ngữ lập trình

Đỗ Đăng Khôi

October 2019

I Phần câu hỏi lập trình

Câu 1

Viết các mô tả thích hợp (token và biểu thức chính qui) trên ANTLR để mô tả token INTLIT, biểu diễn một hằng nguyên thập phân trên ngôn ngữ C. INTLIT là một chuỗi các ký số từ 0 đến 9 nhưng không bắt đầu bằng ký tự số 0 trừ trường hợp bằng 0 (chỉ ghi một số 0).

Ví dụ 1023, 900, 0 là các chuỗi hợp lệ ứng với token INTLIT trong khi các chuỗi sau không hợp lệ: 00 (không bắt đầu bằng số 0 nên chuỗi này sẽ tạo ra 2 token INTLIT(0)), 0123 (không bắt đầu là số 0 nên chuỗi này sẽ tạo 2 token INTLIT(0) và INTLIT(123)), 123a4 (không có ký tự a nên sẽ tạo ra token INTLIT(123) và báo lỗi ở a).

Sinh viên chỉ cần mô tả token INTLIT, không cần xử lý lỗi.

Trả lời : INTLIT : $[1-9]$ $[0-9]*$ | '0'

Câu 2

Viết các mô tả thích hợp (token và biểu thức chính qui) trên ANTLR để mô tả token STRING biểu diễn cho một hằng chuỗi trên ngôn ngữ X. STRING là một chuỗi bắt đầu bằng 2 dấu ", kết thức cũng bằng 2 dấu ", và ở giữa là một chuỗi (có thể rỗng các ký tự bất kỳ nhưng không được phép có 2 dấu " liên tiếp hoặc 1 dấu " nằm ở cuối.

Ví dụ các chuỗi hợp lệ: """abc " de"f"" và "" ??!' abc " mdm" dfg ""

Ví dụ các chuỗi sai "" abc "" "" (chỉ có "" abc "" là tạo thành STRING), "" abc """ (chỉ có "" abc "" là tạo thành STRING).

Trả lời : STRING : '""' (~'"' | '"' ~'"')* '""'

Câu 3

Cho biểu thức có các phép nhị phân @, $\wedge$, và ? trong phép @ có độ ưu tiên cao nhất và có tính kết hợp phải. Phép toán $\wedge$ có độ ưu tiên trung bình và không có tính kết hợp, phép toán ? có độ ưu tiên thấp nhất và có tính kết hợp trái. Toán hạng nguyên tố là số nguyên (INT) và dấu () có thể được dùng để thay đổi độ ưu tiên và tính kết hợp. Hãy viết văn phạm cho biểu thức dùng dạng BNF.

Trả lời :

```
exp0 -> exp0 '?' exp1 | exp1
exp1 -> exp2 '^' exp2 | exp2
exp2 -> exp3 '@' exp2 | exp3
exp3 -> '(' exp0 ')' | INT
```

Câu 4

Một kiểu trên ngôn ngữ TEST có thể là int, float hoặc kiểu array. Một kiểu array trên TEST bắt đầu bằng từ khóa array theo sau là một danh sách miền, kế đến là từ khóa of và cuối cùng là một kiểu. Một danh sách miền bắt đầu bằng dấu '[', kết thức bằng một dấu ']' và ở giữa là các miền cách nhau bằng dấu ','. Phải có ít nhất một miền trong danh sách miền. Mỗi miền được viết bằng 2 số nguyên cách nhau bởi 2 dấu chấm liên tiếp '..'. Một số nguyên có thể là số dương (chỉ gồm các ký tự số) hoặc số âm (bắt đầu bằng dấu '-' theo sau là các ký số).

Ví dụ: array [1..3, −2..4, 15..20] of array [−13..15] of int biểu diễn một array có 3 miền con với kiểu thành phần là một array có 1 miền với kiểu thành phần là int.

Trả lời :

```
arraytype : 'array' '[' domainlist ']' 'of' primtype;
domainlist : domain (COMMA domain)*;
domain : '-'? INTLIT '..' '-'? INTLIT;
primtype : FLOAT | INT | arraytype;
```

II Phần câu hỏi lý thuyết

Câu 5

a. Cho các lớp A,B (lớp con của A), C và Visitor được khai báo như sau:

```
class A:
    def accept(self,v):
        v.visitA(self)
class B(A):
    def accept(self,v):
        v.visitB(self)
class C:
    def accept(self,v):
        v.visitC(self)
```

```
class Visitor:
    def visitA(self,x):
        return x.accept(self)
    def visitB(self,x):
        pass
    def visitC(self,x):
        pass
```

Hỏi lệnh gọi x.accept(self) trong phương thức visitA của class Visitor có thể gọi (gián tiếp) đến các phương thức nào của class Visitor? **Giải thích.**

Trả lời : Vì x(AContext) sẽ gọi hàm accecct của các lớp con của lớp A. Mà ta có B là lớp con của A, nên phương thức được gọi là visitB().

b. Hãy giải thích giải pháp giải quyết "vấn đề kim cương" (diamond problem) trên Python 3 và minh họa cho giải thích trên thông qua xác định chuỗi tìm kiếm lớp cha của đoạn mã Python.

```
class A:
def foo(self):
    print("A")
class B(A):
    pass
class C(B):
    pass
class D(A):
        def foo(self):
            print("D")
```

```
class E(A)
    pass
class F(D,E):
    def foo(self):
        print("F")
class G(C,F,D):
    pass
x = G()
x.foo() #1
```

Trả lời :
L(A) = [A], L(B) = [B, A], L(D) = [D, A], L(E) = [E, A]
L(C) = [C] + merge(L[B], [B]) = [C, B, A]
L(F) = [F] + merge(L(D), L(E), [D, E]) = [F, D, E, A]
L(G) = [G] + merge(L(C), L(F), L(D), [C, F, D]) = [G, C, B, F, D, E, A]
`https://en.wikipedia.org/wiki/C3_linearization`

Câu 6

a. Thế nào là closure trên lập trình hàm? Cho một ví dụ để giải thích?

Trả lời : Closure là một hàm trả về giá trị phụ thuộc vào một hay nhiều biến đã được định nghĩa bên ngoài phạm vi của hàm (closure).

`https://www.tutorialspoint.com/scala/scala_closures.htm`

```
object Demo {
    val factor = 10
    def main(args: Array[String]) {
        return 1 + factor
    }
}
```

b. Viết hàm **subsum** nhận vào một danh sách mà mỗi phần tử là tổng của phần tử tương ứng trong **lst** với tất cả các phần tử đúng trước trong **lst**.

Ví dụ subsum([1,3,4,5]) sẽ trả về [1,4,8,13] trong đó 1 = 1, 4 = 1 + 3, 8 = 1 + 3 + 4, và 13 = 1 + 3 + 4 + 5. Yêu cầu dùng một trong hai phương pháp: dùng đệ quy hoặc dung hàm bậc cao?

Đệ quy

```
def subsum(lst:list) :
    return subsum(list[:-1]) + [sum(lst)] if lst else []
```

Lập trình hàm

```
def subsum(lst:list()) :
    return list(map(lambda i : sum(lst[:i]), range(1,len(lst)))
```