date . . No. HW1

1.4 Having only a single language for all domains need to consider many aspects of implementation, as different domains focus on different features. And that will make the language huge and hard to maintain.

1.18 (1) delimeter used on both side

(a) advantage: comment out an area easily, more writable

(b) disadvantage: less readable

(2) delimeter marks only the beginning

(a) advantage: less possible to make mistake

(b) disadvantage: have to mark each line seperately, inconvinient when you need to mark a block

2.14 (1) for: flexible and brief, easier to apply.

(2) against: the compiler spends less time dynamically typing objects and so execute faster

3.4 <assign> → <id> = <expr>

<id> → A | B | C

<expr> → <expr> + <term> | <term>

<term> → <expr> * <factor> | <factor>

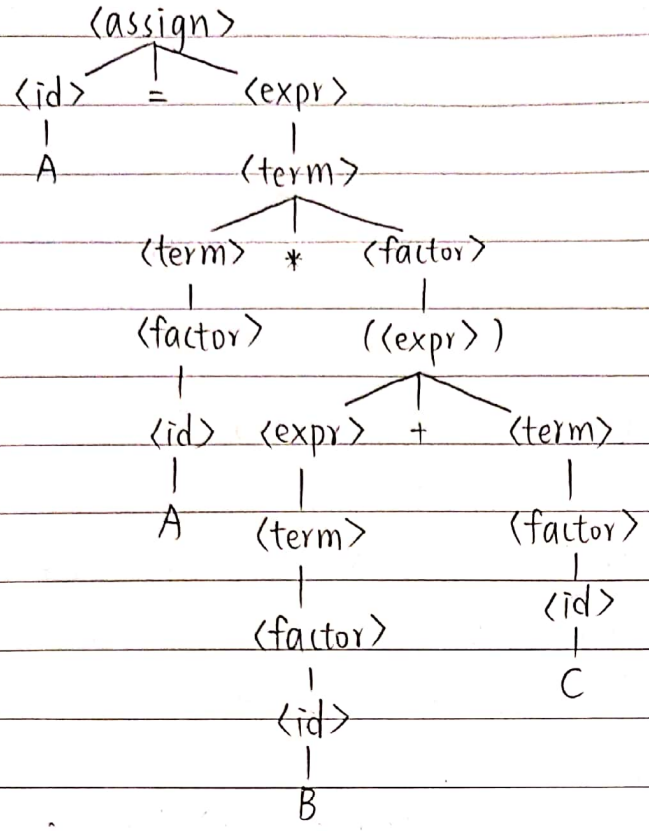
<factor> → (<expr>) | <unary id>

<unary id> → ++<id> | --<id> | <id>++ | <id>--

Ïhý uú

date . . No.

3.7 (c) A = A * (B+C)



<assign> → <id> = <expr>

→ A = <term>

→ A = <term> * <factor>

→ A = <factor> * (<expr>)

→ A = <id> * (<expr> + <term>)

→ A = A * (<term> + <factor>)

→ A = A * (<factor> + <id>)

→ A = A * (<id> + C)

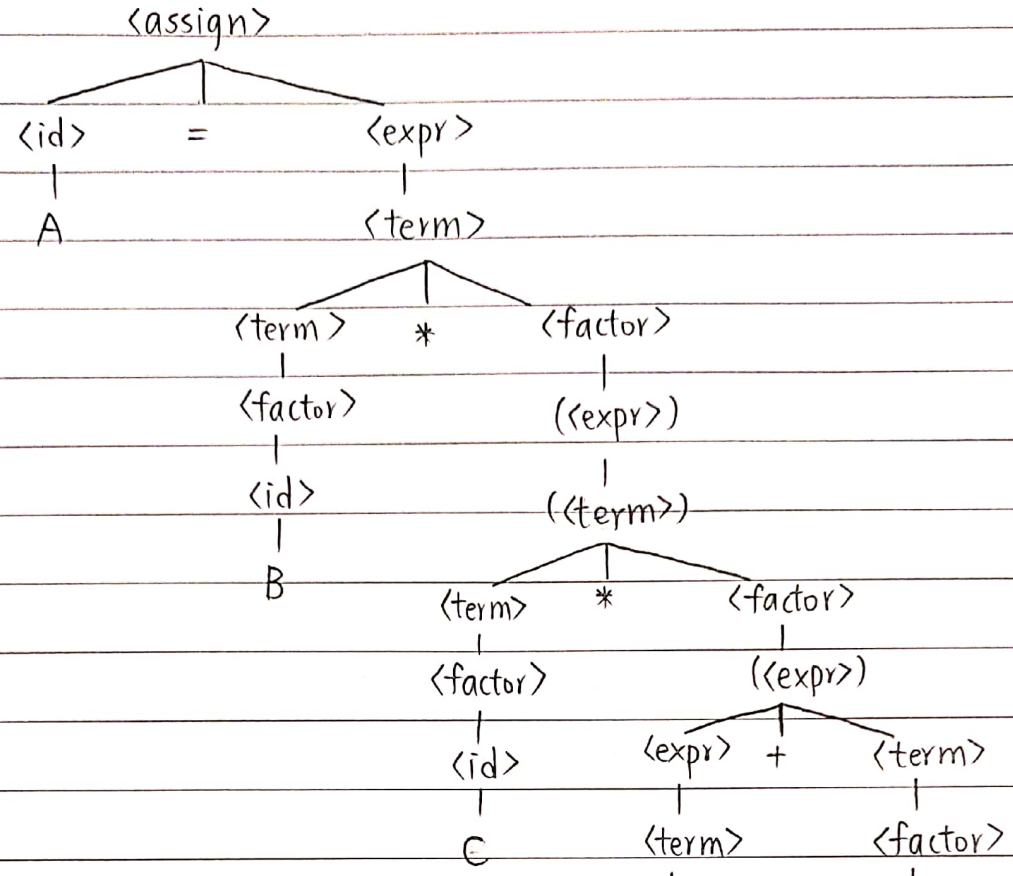
→ A = A * (B+C)

A

Ïhý uú

date . . No.

3.7(d) A=B * (C * (A+B))



<assign> → <id> = <expr>

→ A = <term>

→ A = <term> * <factor>

→ A = <factor> * (<expr>)

→ A = <id> * (<term>)

→ A = B * (<term> * <factor>)

→ A = B * (<factor> * (<expr>))

→ A = B * (<id> * (<expr> + <term>))

→ A = B * (C * (<term> + <factor>))

→ A = B * (C * (<factor> + <id>))

→ A = B * (C * (<id> + B))

→ A = B * (C * (A+B))

Ïhý uú

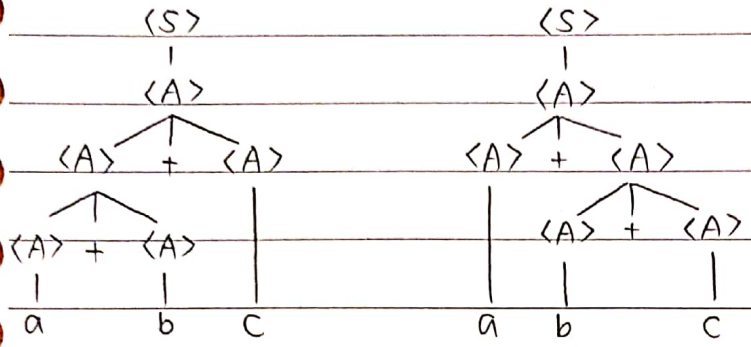
date . . No.

3.8 ⟨S⟩ → ⟨A⟩

⟨A⟩ → ⟨A⟩+⟨A⟩ | ⟨id⟩

⟨id⟩ → a | b | c

same string have different parse tree :



3.11 ⟨S⟩ → ⟨A⟩a⟨B⟩b

⟨A⟩ → ⟨A⟩b | b ⇒ (a) babb

⟨B⟩ → b (b) bbbabb

3.13 ⟨S⟩ → ⟨A⟩b

⟨A⟩ → a⟨A⟩b | ab

3.23 (b) b=(C+10)/3 {b>6} (c) a=a+2*b-1 {a>1}

⇒ (C+10)/3 > 6 a+2*b-1 > 1

C+10 > 18 2b > 2-a

C > 8 $b > \frac{2-a}{2}$

5.6 (a) (i) sub1 : sub1 (b) (i) sub1 : sub1

(ii) sub2 : sub1 (ii) sub2 . sub1

(iii) sub3 : main (iii) sub3 : sub1

dynamic scoping 是用最新 define 的

5.8 sub1: a → sub1 sub2: a → sub2 sub3: a → sub3

y → sub1 b → sub2 x → sub3

z → sub1 z → sub2 w → sub3

x → main y → sub1 y → main

x → main z → main

GEE-JUMP

Ïhý uú