Programming Languages HW 1

A10515003

二資工三

鄧鵬宇

1.4 每一種程式語言的特性和使用範圍都是不一樣的，用一種程式語言去模擬另一種程式語言并不是不可能的事情，但是會花費額外的成本，時間和精力，只用一種語言去解決所有程序問題是非常沒有效率的。程式語言被發明出來是爲了解決特定的問題，比如PHP和C語言針對的領域是不一樣的，不太會有人想用C語言去做web開發，這時候針對實際環境去選擇正確的程式語言才是正確的方式，而不是用一門語言嘗試解決所有問題。

1.18 單行注釋的優點是，使用字符少，適合快速少量的注釋内容，只針對單行生效，不會影響到其他範圍。缺點是不適合注釋大量内容。

多行注釋的優點是，可以用較少的字符注釋大量内容。比較適合成段多行内容注釋。缺點是如果少量注釋便會顯得很繁瑣。如果程序中含有多個注釋，此時缺少開始或者結束符會造成中間代碼部分被錯誤注釋。

2.14 對於無類型的支持者來説，無類型的好處有，比較靈活和簡潔的語法。對於反對者來説，無類型造成無法使用類型檢查在不執行程式時驗證數據的完整性。而且使用有類型的語言可以使執行效率更快。

3.4 <assign> -> <id> = <expr>

<id> -> A | B | C

<expr> -> <expr> + <term> | <term>

<term> -> <term> \* <factor> | <factor>

<factor> -> (<expr>) | <id> | <id> ++ | <id> - -

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

<assign> => <id> = <expr>

=> A = <expr>

=> A = <term> \* <factor>

=> A = <factor> \* <factor>

=> A = <id> \* <factor>

=> A = A \* <factor>

=> A = A \* (<expr>)

=> A = A \* (<expr> + <term>)

=> A = A \* (<term> + <term>)

=> A = A \* (<factor> + <term>)

=> A = A \* (<id> + <term>)

=> A = A \* (B + <term>)

=> A = A \* (B + <factor>)

=> A = A \* (B + <id>)

=> A = A \* (B + C)

<assign>

<id> = <expr>

A <term> \* <factor>

(expr)

<factor>

<expr> + <term>

<id>

<term> <factor>

A

<factor> <id>

<id> C

B

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

<assign> => <id> = <expr>

=> A = <term> \* <factor>

=> A = <factor> \* <factor>

=> A = <id> \* <factor>

=> A = B \* <factor>

=> A = B \* (<expr>)

=> A = B \* (<term>)

=> A = B \* (<term> \* <factor>)

=> A = B \* (<factor> \* <factor>)

=> A = B \* (<id> \* <factor>)

=> A = B \* (C \* <factor>)

=> A = B \* (C \* (<expr>))

=>A = B \* (C \* (<expr> + <term>))

=>A = B \* (C \* (<term> + <term>))

=>A = B \* (C \* (<factor> + <term>))

=>A = B \* (C \* (<id> + <term>))

=>A = B \* (C \* (A + <term>))

=>A = B \* (C \* (A + <factor>))

=>A = B \* (C \* (A + <id>))

=>A = B \* (C \* (A + B))

<assign>

<id> = (<expr>)

<term> \* factor

A

<factor> (<expr>)

<id> (<term>)

B <term> <factor>

<factor> (<expr>)

<id> <expr> <term>

C <term> <factor>

<factor> <id>

<id> B

A

3.8 如果是A+B+C的話會是下面所示的有兩種parse tree表示

<S>

<A>

<A> + <A>

<id> <A> + <A>

A <id> <id>

B C

Type-A

<S>

<A>

<A> + <A>

<A> + <A>

<id>

<id> <id>

C

A B

Type-B

3.11 11. Consider the following grammar:

<S> → <A> a <B> b

<A> → <A> b | b

<B> → b

Which of the following sentences are in the language generated by this

grammar?

a.babb b. bbbabb

3.13 <A> -> ab | a <A> b

3.23 b. b = (c + 10) / 3 { b > 6 }

( ( c + 10 ) / 3 ) > 6

c + 10 > 18

c > 8

C a = a + 2 \* b – 1 { a > 1 }

a + 2 \* b – 1 > 1

2 \* b > 2 – a

b > ( 2 - a ) / 2

5.6 a i. sub1 ii. sub1 Iii. main

b i. sub1 ii. sub1 Iii. sub1

5.8 Sub1: sub1(a,y,z) main(x)

Sub2: sub1(y) sub2(a,b,z) main(x)

Sub3: sub3(a,x,w) main(y,z)