Simple Java － Scanner

B113040052 陳育霖

**Lex 版本**

使用版本：**flex 2.6.4**

**作業平台**

作業系統：**Ubuntu 22.04.4 LTS**

編譯器：**gcc (Ubuntu 11.4.0-1ubuntu1~22.04) 11.4.0**

工具鏈：flex、gcc、make

**執行方式**

1. **使用make指令自動編譯：**

make

1. **會生成執行檔demo**

執行 demo並輸入測試檔：

./demo < test.java

1. **結果會輸出每個token的：**

類型（如：ID、float、operator）

所在的行號

該行的第幾個字元

例如：Line: 1, 1st char: 1, "// print hello world" is a "comment".

1. **最後輸出Symbol Table中所有的識別字（Identifier）**

例如：

The symbol table contains:

a

**處理規格書上的方式**

**Token 的分類與判斷處理：**

為符合規格書所提的分類需求，我將Scanner所辨識的token分為以下幾類：

**Reserved Words（保留字）**：以字串表的方式將所有reserved word儲存（如 int, float, if, else…等），直接在Lex中用條件語法判斷是否屬於保留字。

**Identifiers（識別字）**：用正規表示式[a-zA-Z\_][a-zA-Z0-9\_]\*，在辨識到後檢查是否為保留字，是的話則分類為 reserved word，否則儲存至symbol table。

**整數（integer）**：用 [0-9]+ 處理，額外加入負號支援（例如：-123）。

**浮點數（float）：**處理 1.0, 3.14, -0.5, 12.5e+3, -1.2E-5…等格式，使用多組 regex 結合，並加上狀態切換避免誤判。

**String（字串常數）**：正規表示式處理雙引號包住的內容，並支援 \" 內嵌的雙引號，例如："aa\"bb"。

**Operators & Symbols（運算子與符號）**：個別設立規則對應所有符號與運算子（如：+, ++, =, <=, !=, (, )…等），用 Lex 的固定字串處理。

**Comments（註解）**：

**單行註解： //**：掃描至該行結尾為止。

**多行註解： /\* ... \*/**：使用start condition處理多行狀態，可處理註解內嵌註解結構，不跨越行未結束註解會提示錯誤。

**錯誤處理與 recovery**：

遇到非法字元時會輸出錯誤訊息，並跳過繼續處理後續token。

例如：Line: 1, 1st char: 1, "1a" is a "error".

**Symbol Table的建立與管理**：

hash = (hash \* 33 + c) % table\_size；c = 該名稱之字元。

create()：初始化 symbol table。

lookup(char\* s)：檢查字串是否已存在，存在回傳 index，否則回傳 -1。

insert(char\* s)：若字串不存在，加入 symbol table。

dump()：列印所有已儲存的識別字。

**遇到的問題與解決方式**

1. 遇到無內容的換行會一直被判定為error

解決方法：加上\r (Carriage Return) \v (垂直的tab) \f (換頁) 的判斷

1. 字串會將開頭與結尾的雙引號"讀取並輸出  
   解決方法：將所有字元往前移一格，並將倒數第二個字元改成\0

**測試檔執行出來的結果**

Test1.java

**一張含有 文字, 螢幕擷取畫面, 功能表, 設計 的圖片

AI 產生的內容可能不正確。一張含有 文字, 螢幕擷取畫面, 功能表 的圖片

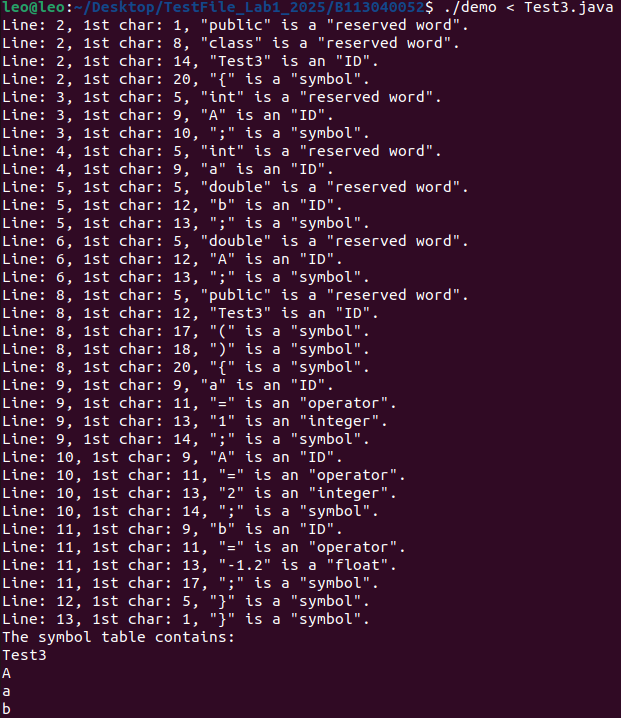
AI 產生的內容可能不正確。**

Test2.java

**一張含有 文字, 螢幕擷取畫面 的圖片

AI 產生的內容可能不正確。**

Test3.java

****