**Scanner**

### **Description**

Token is the smallest unit in a compiler. Input text should be transformed into token by scanner first, then Parser can do the next step. Please write a scanner to get tokens by Table 1. and print it out.

**Input Format**

Input one line program source, each line will end with a ‘\n’ character.

Not all input will follow the token definition.

### **Output Format**

If the program source follow the definition, print each token’s type and the string of token seperated by a whitespace and end with a newline.

Otherwise print only “invalid input” with a newline even if there is just one wrong. (don’t output any token!)

Table 1.

|  |  |
| --- | --- |
| **Terminal** | **Regular Expression** |
| ID | [A-Za-z\_][A-Za-z0-9\_]\* |
| STRLIT | “[^”]\*” |
| LBR | \( |
| RBR | \) |
| DOT | \. |
| SEMICOLON | ; |

|  |  |
| --- | --- |
| **Sample Input1**  Str.length("123");  **Sample Output1**  ID Str  DOT .  ID length  LBR (  STRLIT "123"  RBR )  SEMICOLON ; | **Sample Input2**  Str.length("123");  1  **Sample Output2**  invalid input |
| **Sample Input3**  "string;  **Sample Output3**  invalid input | **Sample Input4**  "test\_string";  Test\_ID;  **Sample Output4**  STRLIT "test\_string"  SEMICOLON ;  ID Test\_ID  SEMICOLON ; |

In Sample Output2, although “Str.length("123");” can be cut by definition correctly, but there is no definition for “1”, so print “invalid input”.

In Sample Output3, “STRLIT” token should have two “ " ”, there is no definition for only one “ " ”, so print “invalid input”.