**Grammar**

### **Description**

Refer to Table 1, please modify the program according to the solution in Problem 1. Your program has to check the source whether it follows grammar rules or not. If yes, print “Valid” with a newline. If no, print “Invalid input” with a newline.

**You will get no credits if you are not using Recursive-Decent-Parsing.**

### **Input Format**

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

Not all input will follow the token definition and grammar rule.

### **Output Format**

If the program source follow the rule, print “Valid” with a newline.

If the program source not follow the rule, print “Invalid” with a newline.

Table 1.

|  |
| --- |
| **Productions**  1 program → stmts  2 stmts → stmt stmts  3 stmts → λ  4 stmt → exp SEMICOLON  5 exp → primary  6 exp → STRLIT  7 exp → λ  8 primary → ID primary\_tail  9 primary\_tail → DOT ID primary\_tail  10 primary\_tail → LBR exp RBR primary\_tail  11 primary\_tail → λ |

|  |  |
| --- | --- |
| **Sample Input** "test\_string";  Test\_ID; **Sample Output** Valid | **Sample Input** “string”; **Sample Output** Valid |
| **Sample Input** Str.length(; **Sample Output** Invalid | **Sample Input** Str.; **Sample Output** Invalid |