컴파일러구성

– 제5강 어휘분석과 LEX

김강현 교수

목차

어휘분석이란?

원시프로그램을 읽어 들여 토큰이라는 의미있는 문법 단위로 분리하는 것.

정규문법이 주어지고 문법단위에 대한 토큰표를 작성한 후 DFA를 작성한다.

어휘 분석의 예

- 원시프로그램

```
ABC := E * 3.14 + ABC/E;
```

- 어휘분석 결과 → 10개의 토큰

```
ABC   :=   E   *   3.14
  +  ABC   /   E     ;
```

어휘 분석의 예

- 원시프로그램

```
ABC := E * 3.14 + ABC/E;
```

- 어휘분석 결과 → 10개의 토큰

```
ABC   :=   E   *   3.14
  +  ABC   /   E     ;
```

→

```
(1,0)   :=   (1,1)  *   (2,0)
 +     (1,0)    /   (1,1)     ;
```

◆ 토큰의 종류

(1) 식별자(identifier) :프로그래머가 정의하는 변수

(2) 상수(constant): 정수형, 실수형, 문자형 상수

(3) 예약어(reserved word) :

이미 정의된 지정어, DO, IF, WHILE

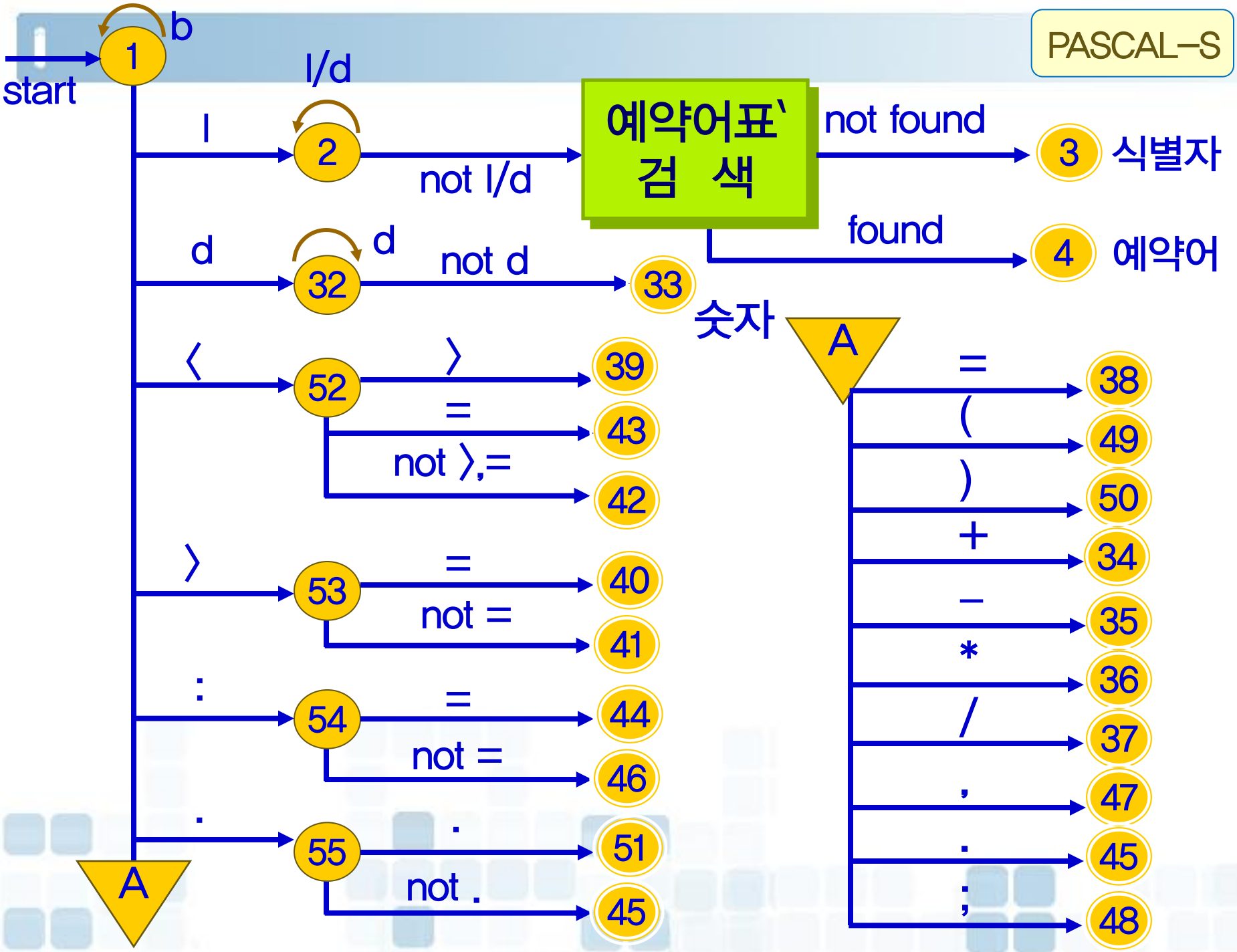
(4) 연산자(operator): –, +, *, / 등

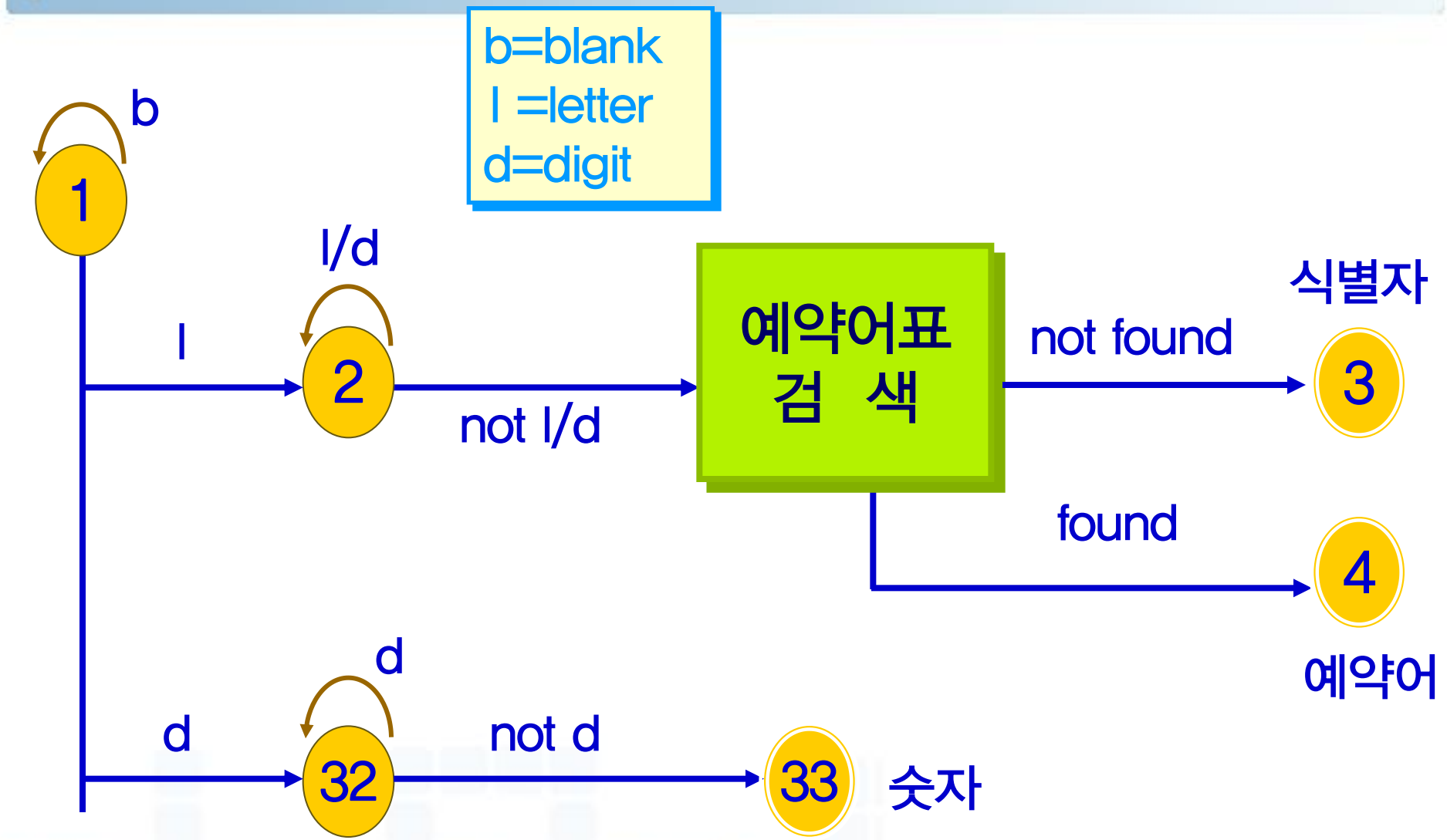
(5) 구분자(delimiter) : (, [, ;, :, 등 단어를 구분하는 기회

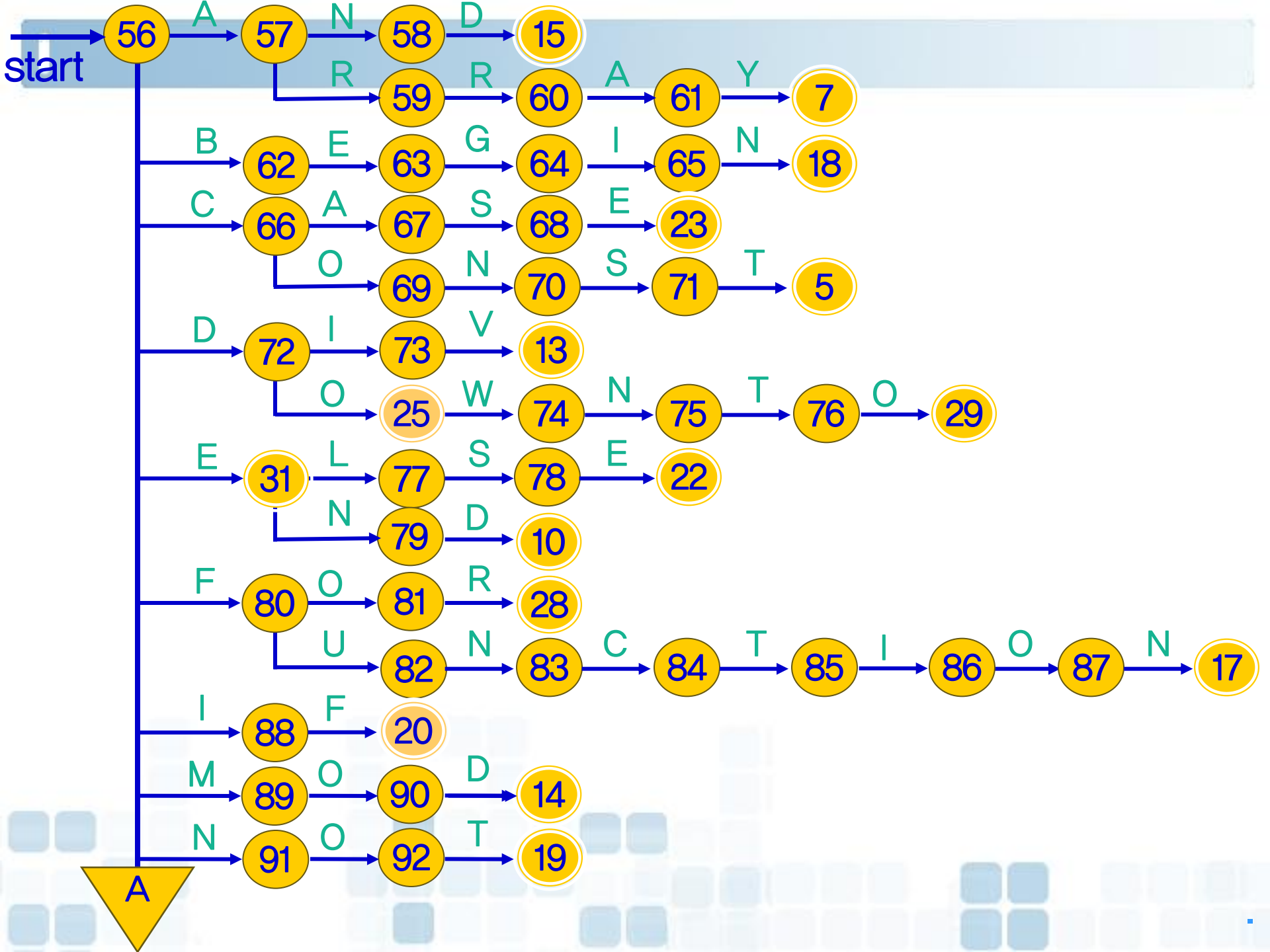
어휘 분석기 설계

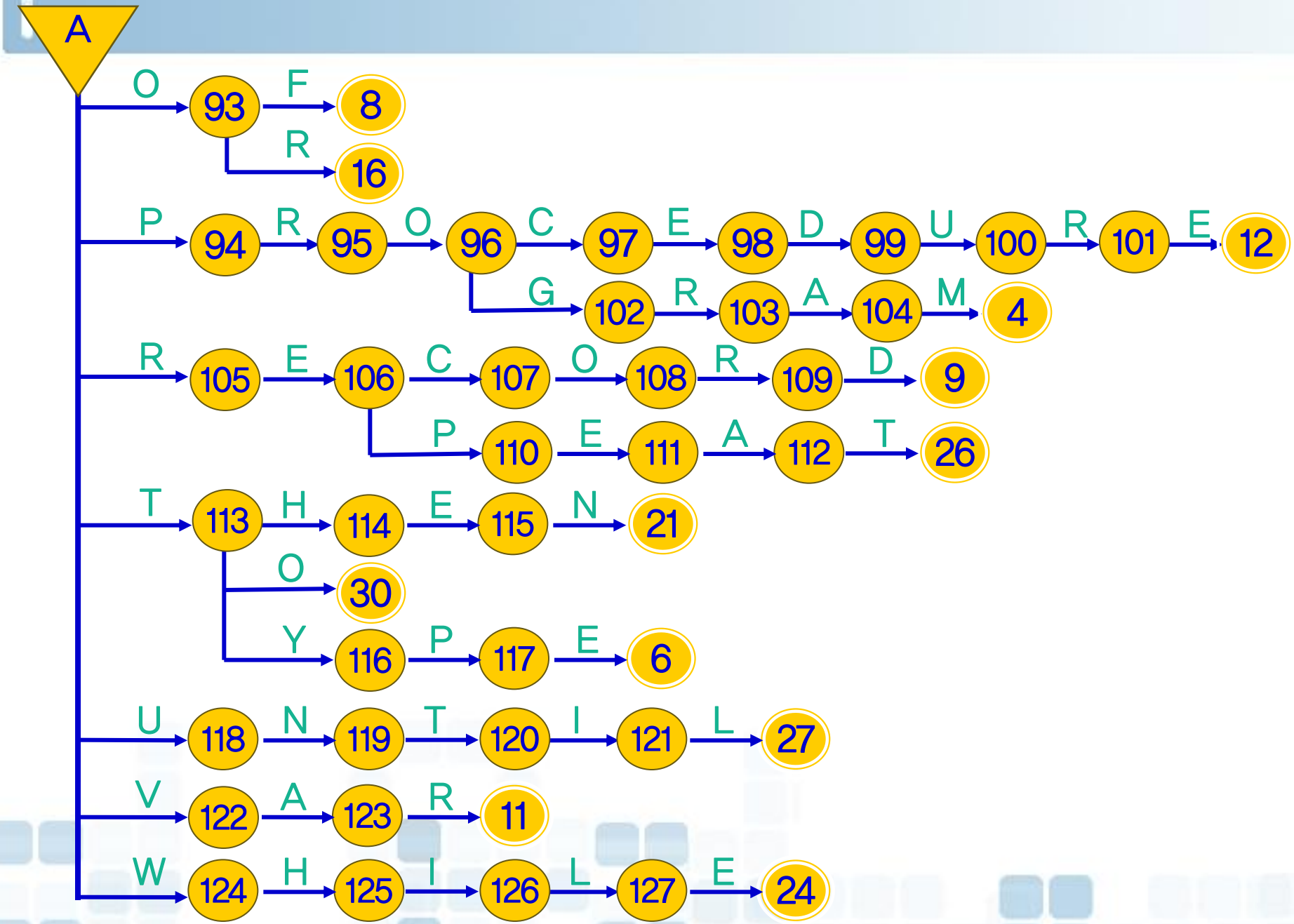
토큰표

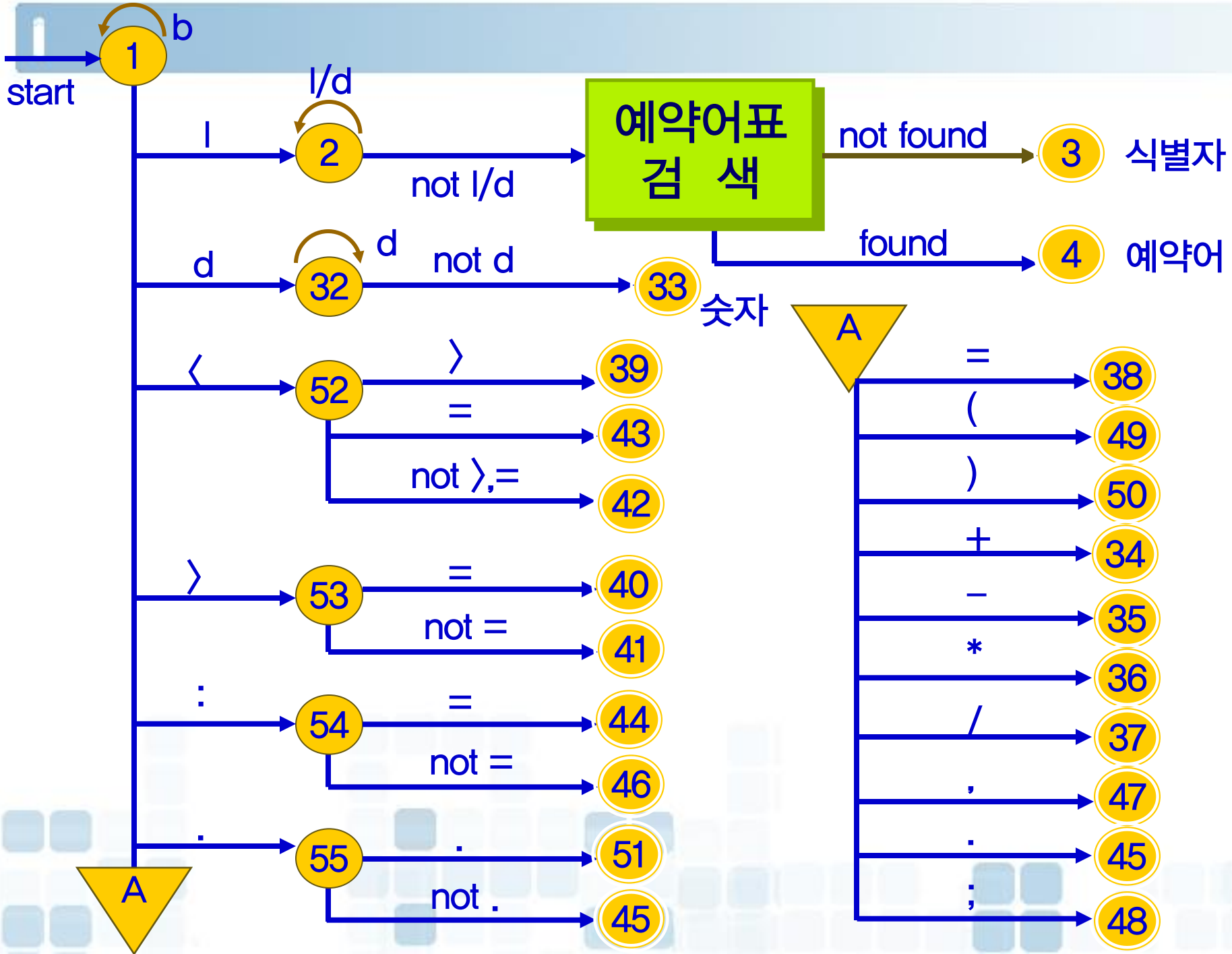
토큰이름	토큰번호	토큰값
BEGIN	1	
CONST	2	
END	3	
식별자	4	기호표의 포인터
상수	5	기호표의 포인터
=	6	
;	7	


start
1
b
l/d
l
2
not l/d
예약어표`
검 색
not found
3
식별자
found
4
예약어
d
32
d
not d
33
숫자
A
〈
52
〉
39
=
43
not 〉,=
42
〉
53
=
40
not =
41
:
54
=
44
not =
46
.
55
.
51
not .
45
A
=
38
(
49
)
50
+
34
−
35
*
36
/
37
,
47
.
45
;
48

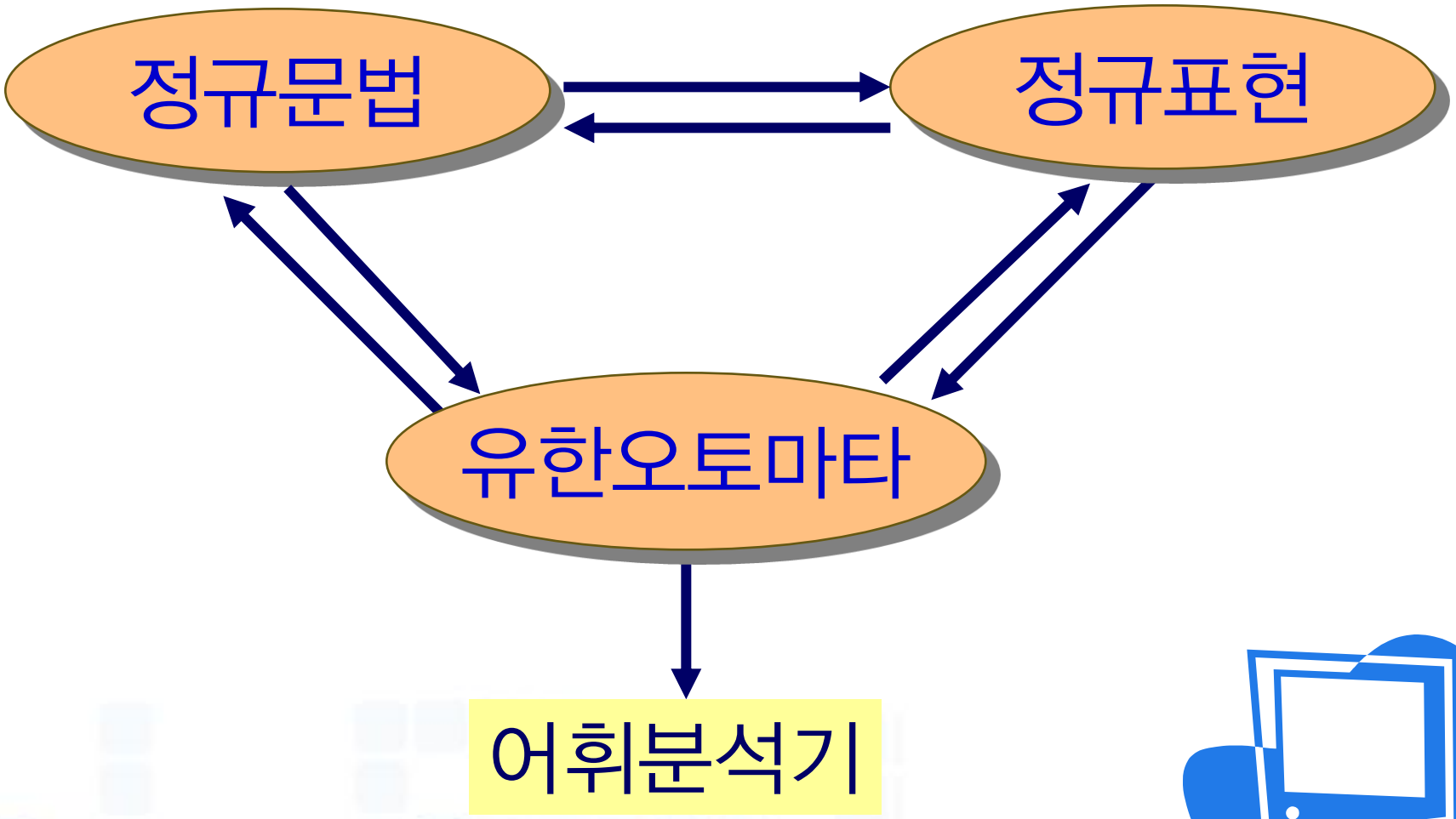

b=blank
l =letter
d=digit
b
1
l/d
l
2
not l/d
예약어표
검 색
not found
식별자
3
found
4
예약어
d
d
32
not d
33
숫자


start
56
A
57
N
58
D
15
R
59
R
60
A
61
Y
7
B
62
E
63
G
64
I
65
N
18
C
66
A
67
S
68
E
23
O
69
N
70
S
71
T
5
D
72
I
73
V
13
O
25
W
74
N
75
T
76
O
29
E
31
L
77
S
78
E
22
N
79
D
10
F
80
O
81
R
28
U
82
N
83
C
84
T
85
I
86
O
87
N
17
I
88
F
20
M
89
O
90
D
14
N
91
O
92
T
19
A


A
O
93
F
8
R
16
P
94
R
95
O
96
C
97
E
98
D
99
U
100
R
101
E
12
G
102
R
103
A
104
M
4
R
105
E
106
C
107
O
108
R
109
D
9
P
110
E
111
A
112
T
26
T
113
H
114
E
115
N
21
O
30
Y
116
P
117
E
6
U
118
N
119
T
120
I
121
L
27
V
122
A
123
R
11
W
124
H
125
I
126
L
127
E
24


start
1
b
l/d
l
2
not l/d
예약어표
검 색
not found
3
식별자
found
4
예약어
d
32
d
not d
33
숫자
A
〈
52
〉
39
=
43
not 〉,=
42
〉
53
=
40
not =
41
:
54
=
44
not =
46
.
55
.
51
not .
45
A
=
38
(
49
)
50
+
34
–
35
*
36
/
37
,
47
.
45
;
48

동치관계



어휘분석기 구현 고려사항

- 문법이 명확히 정의되어야 한다.
- 상태전이도 순서가 중요하다:

  많이 사용되는 토큰을 구별해야(확률개념)
- 구문분석기와 어휘분석기의 호출순서
- 실수의 내부표현 문제
- 문자 검조 시간 단축 문제

컴파일러 구현

◈ 언어와 목적기계(하드웨어)가 발달할수록
많은 컴파일러가 필요하다.

◈ 예를 들어,
N개의 언어를 M개의 컴퓨터에서 구현하려면
N*M개의 컴파일러가 필요하다.

컴파일러 구현

- 2개의 언어 : Java, C
- 3개의 목적기계 : VAX, IBM, Intel 80586

다음과 같은 6개의 컴파일러가 필요하다.

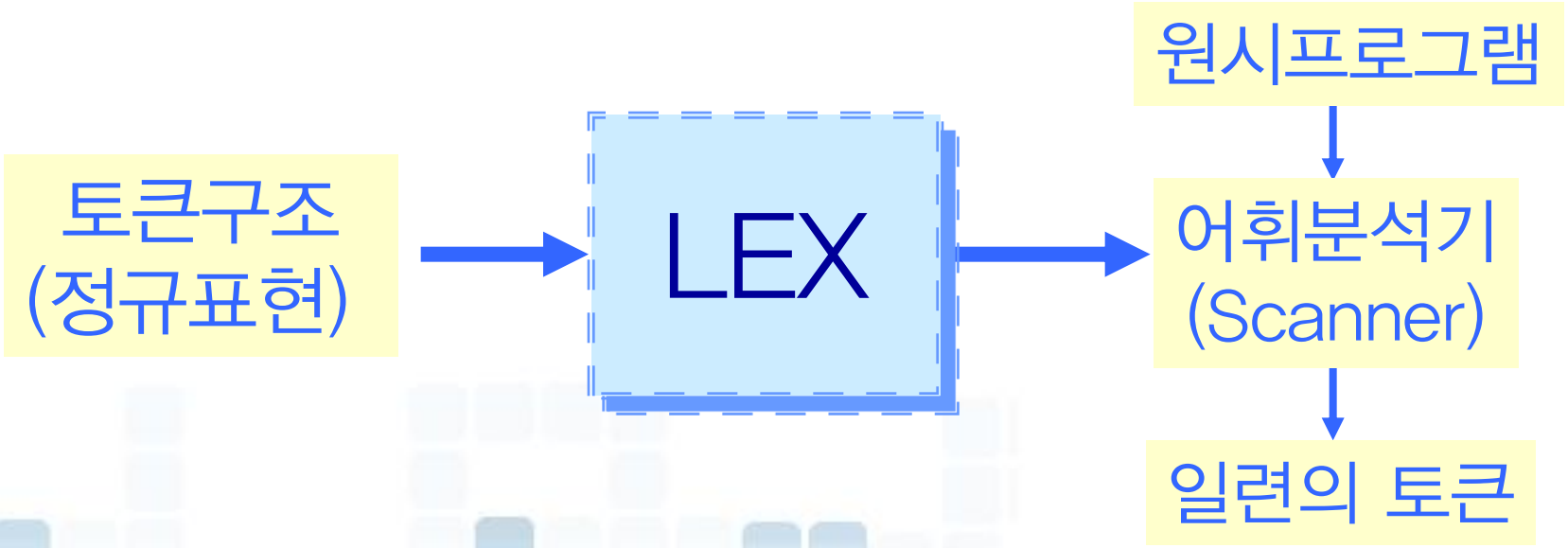
- Java–to–VAX
- Java–to–IBM
- Java–to–Intel 80586
- C –to–VAX
- C –to–IBM
- C –to–Intel 80586

어휘분석기 구현방법

- 프로그래밍 언어를 이용하여 직접 구현
- 컴파일러 자동화 도구를 이용하여 구현
  - 어휘분석기 생성기 종류
    - LEX
    - FLEX
    - ScanGen
    - PCLEX
    - FOISON
    - JLEX

어휘분석기 생성기

LEX:1975년에 *M.E.Lesk*가 고안.
토큰을 구분하는 프로그램 작성도구.

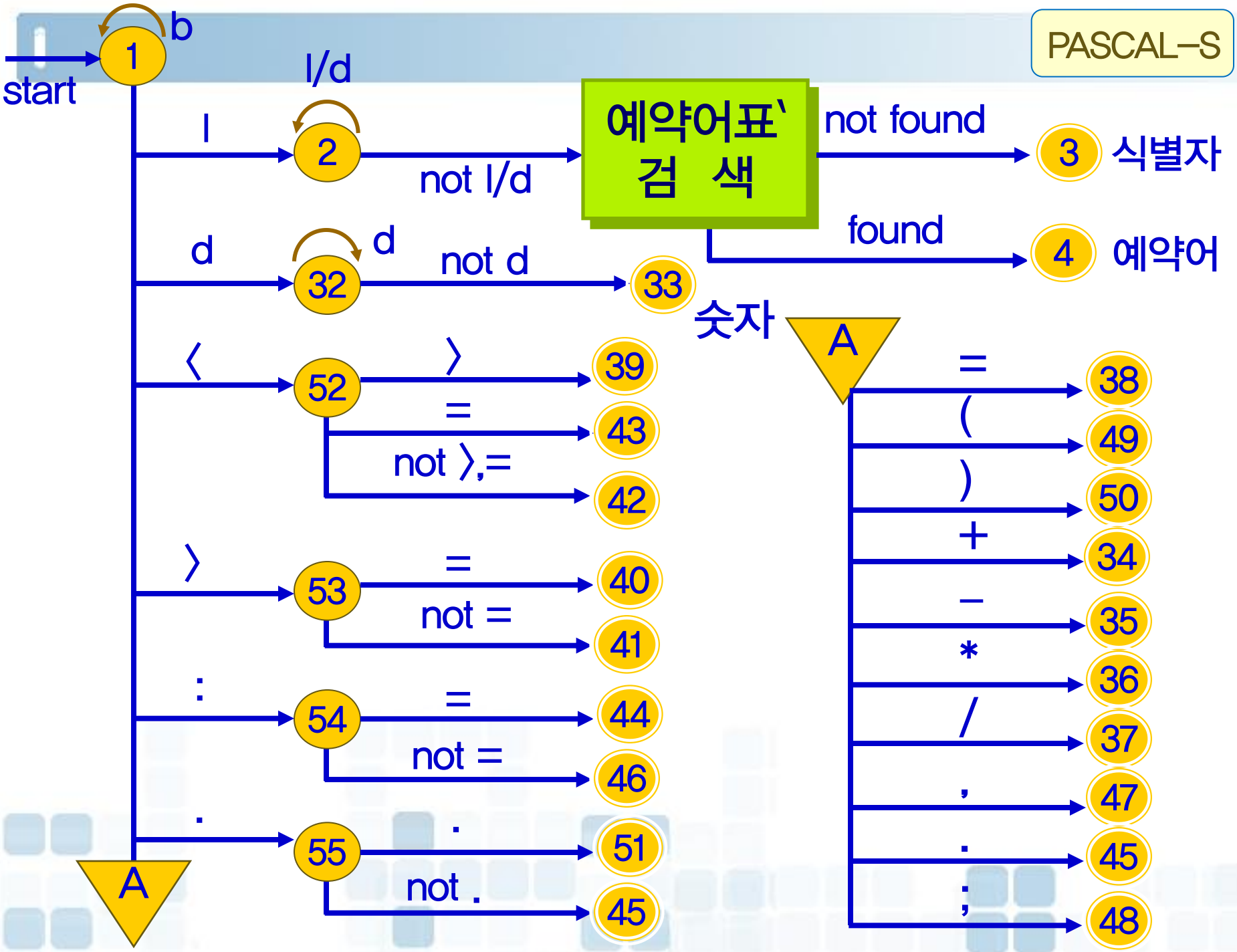


질문코너1

토큰표

토큰이름	토큰번호	토큰값
BEGIN	1	
CONST	2	
END	3	
식별자	4	기호표의 포인터
상수	5	기호표의 포인터
=	6	
;	7	

질문코너2


start
1
b
l/d
l
2
not l/d
예약어표`
검 색
not found
3
식별자
found
4
예약어
d
32
d
not d
33
숫자
A
〈
52
〉
39
=
43
not 〉,=
42
〉
53
=
40
not =
41
:
54
=
44
not =
46
.
55
.
51
not .
45
A
=
38
(
49
)
50
+
34
−
35
*
36
/
37
,
47
.
45
;
48

수고하셨습니다.

곧이어 제2부를 학습하시겠습니다.