Complier Design
Assignment - Unit 2

Submitted By:

Habibur Rahman

17BCS071

Submitted To:

Mr. Sarfaraz Masood

B-Tech VIth Semester (Computer Science)

Department of Computer Engineering,

Faculty of Engineering and Technology,

Jamia Millia Islamia ,

New Delhi

Session: 2019-20

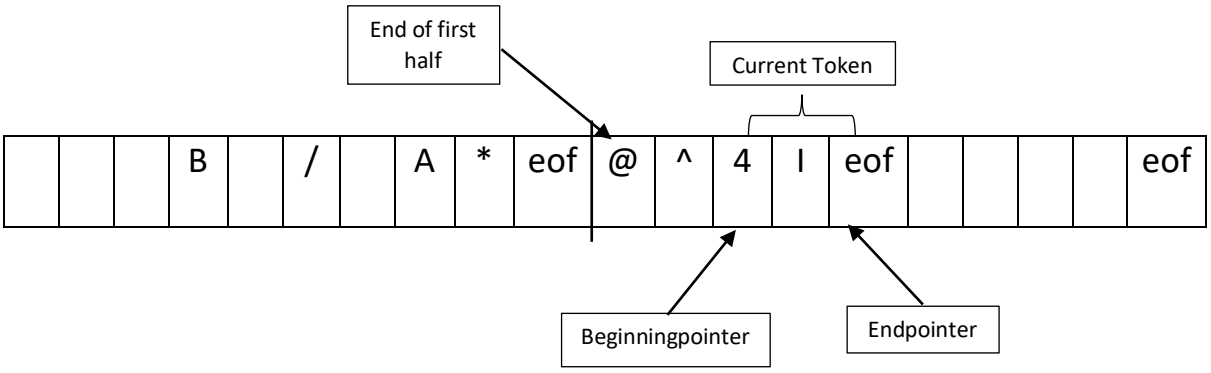
Q1. Explain the sentinel buffer pair algorithm for buffer management. Also exemplify the benefits of maintaining two buffers for the scanner.

Soln.)

In one buffer scheme, only one buffer is used to store the input string . if the lexeme is short then there is no problem but if the lexeme is large then it acroases the buffer boundary and hence overwriting problem occurs .

A large source program consists of a large number of characters to be processed at compile time. Thus, the amount of time taken in compilation is significant. Hence, to reduce the time taken for processing these characters, new techniques have been adopted that help reduce the overhead of processing characters. One such technique uses two buffers instead of one. These buffers are loaded alternately. Both buffers are of same size. We consider this size to be N. Thus, N characters can be read by a single buffer with system read command. A special character ‘eof’ is used to denote the end of the source file. Also, we use two pointers: one is the Beginningpointer (denotes beginning of current token) while the other is the Endpointer (denotes end of current token). The Endpointer keeps scanning for a matching pattern.

When a lexeme is selected, Endpointer points to the last character. Next, after processing of the present lexeme, Beginningpointer moves to the first character of the next lexeme. Before moving forward we have to check whether the end of the buffer is reached. If we have, the second buffer is reloaded through the input.



Algorithm:

```
While (1)
{
        Endpointer = Endpointer + 1 ;
        if (Endpointer is at eof) then
                if (Endpointer is at end of first half) then
                        re-load second half ;
                        Endpointer = Endpointer + 1 ;
                        END
                else if (Endpointer is at end of second half ) then
                        re-load first half ;
                        move Endpointer to the start of first half
                END
        else
                END LEXICAL ANALYSIS
        END
}
```

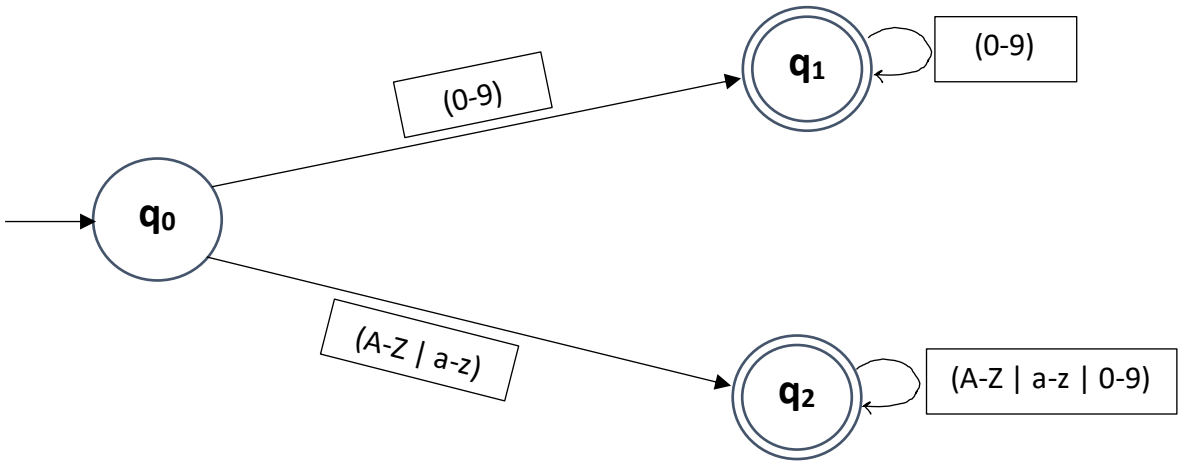
Q2. Construct a DFA for a scanner which would accept only identifiers and numbers in the system.

Soln.)

id -> (A-Z | a-z) (A-Z | a-z | 0-9)*

number -> (0-9)*

q0 is the initial state and **q1** and **q2** are the final states



DFA for scanners accepting only identifiers and numbers