Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Compiler Theory: CS31003
3rd year CSE, 5th Semester

Theory Quiz *Marks: 25*
Date: September 24, 2020

1. Identify the number of tokens in the following C statement. [2]

```
printf("k = %d, &k = %d", k, &k);
```

a) 3
b) 10
c) 22
d) 26

Answer: b)

```
printf
(
"k = %d, &k = %d"
,
k
,
&
k
)
;
```

2. Select the attributes which define Python Typing. [1]

a) Strong
b) Weak
c) Dynamic
d) Static

Answer: a), c)

3. How many minimum registers are required to convert the following intermediate code to its optimised version? [2]

```
t8  =  floattoint(60.5)
t7  = id1 * t8
t4  = id3 + t7
id4 = t4
```

a) 2

b) 1

c) 0

d) Not possible to produce any optimised version by reducing temporary registers

Answer: b)

```
t7 = id1 * 60
id4 = id3 + t7
```

4. The _____ Analyzer phase parses the tokens to create intermediate tree representations. (one word answer, write in small letters, do not capitalise the first letter also) [2]

   Answer: syntax

5. Consider the intermediate code below

```
R1 = y
R2 = z
R1 = R1 - R2
x = R1
```

   The target code for the above is given below. Fill in the blank with the correct instruction (s). [2]

```
LD R1, y
LD R2, z
-----------------
ST x, R1
```

   a) `SUB R2, R1, R2`

   b) `SUB R1, R2`
      `LD R1`

   c) `SUB R2, R1`

   d) `SUB R1, R1, R2`

   Answer: d)

6. Consider the following flex specification

```
%%
a*b           printf("1");
(a|b)*b       printf("2");
c*            printf("3");
```

What is the output for input string `cbbbbac`? [2]

a) `32`

b) `322223`

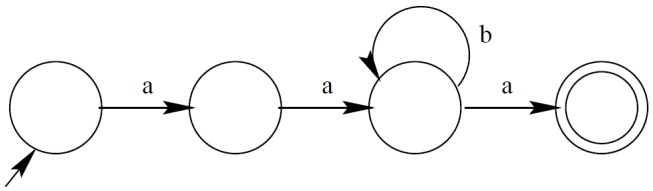
c) `32a3`

d) `323`

Answer: c)

The result is 32a3. The tokenization is c bbbb a c, where the single character 'a' does not match any regular expression and is thus echoed back to the console.

7. Consider the following DFA



Which of the strings can be accepted by the DFA? [1]

a) aab*a

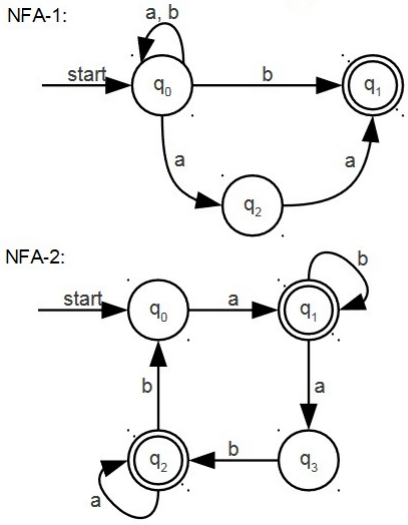
b) aaabbaa

c) aaba

d) aaabaaa

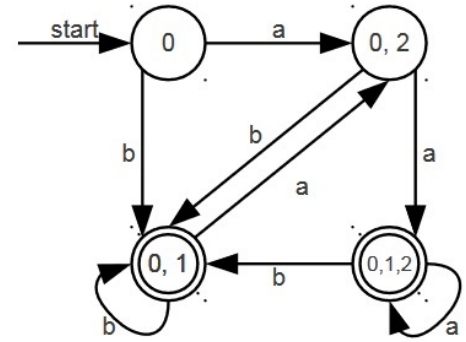
Answer: a), c)

8. Consider the two NFA's below

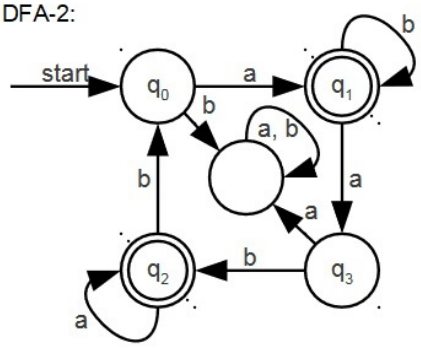


The corresponding DFA's are provided below

DFA-1:



DFA-2:



Identify the correct statement. [2]

(a) NFA-1 is equivalent to DFA-1, NFA-2 is not equivalent to DFA-2

(b) NFA-1 is not equivalent DFA-1, NFA-2 is equivalent to DFA-2

(c) NFA-1 is not equivalent DFA-1, NFA-2 is not equivalent DFA-2

(d) NFA-1 is equivalent to DFA-1, NFA-2 is equivalent to DFA-2

Answer: d)

9. Identify correct regular expression for a C identifier. [2]

a) `[a-z_A-Z][0-9a-z_A-Z]`*

b) `[0-9a-z_A-Z]`$^+$

c) `[a-zA-Z][0-9a-zA-Z]`*

d) `[a-z_A-Z]`$^+$

Answer: a)

10. We need to write a flex program which will tokenize keyword, identifier and integer literal present in a C program. Select the order in which we need to place the three flex rules so that tokenization can be done correctly, without any conflict. [2]

a) `integer literal, identifier, keyword`

b) `integer literal, keyword, identifier`

c) `keyword, identifier, integer literal`

d) `identifier, keyword, integer literal`

Answer: b), c)

11. Select the grammar which is not left recursive. [2]

a)
$$S \rightarrow AB$$
$$A \rightarrow Aa \mid b$$
$$B \rightarrow c$$

b)
$$S \rightarrow Ab \mid Bb \mid c$$
$$A \rightarrow Bd \mid \varepsilon$$
$$B \rightarrow e$$

c)
$$S \rightarrow Aa \mid B$$
$$A \rightarrow Bb \mid Sc \mid \varepsilon$$
$$B \rightarrow d$$

d)
$$S \rightarrow Aa \mid Bb \mid c$$
$$A \rightarrow Bd \mid \varepsilon$$
$$B \rightarrow Ae \mid \varepsilon$$

Answer: b)

12. Is the following grammar ambiguous?
S -> SS|AB , A -> Aa|a , B -> Bb|b [1]

(a) TRUE

(b) FALSE

Answer: a)

13. consider the following grammar
E -> E * F | F+E | F
F -> F-F | id
Identify the correct statement [2]

(a) * has higher precedence than +

(b) – has higher precedence than *

(c) + and — have same precedence

(d) + has higher precedence than *

Answer: b)

14. A shift reduce parsing table and the corresponding grammar is given below.

1) E -> E + T
2) E -> T
3) T -> (E)
4) T -> id

State on top of stack	Action					Goto	
	id	+	(	)	$	E	T
0	s4		s3			1	2
1		s5			accept		
2	r2	r2	r2	r2	r2		
3	s4		s3			6	2
4	r4	r4	r4	r4	r4		
5	s4		s3				8
6		s5		s7			
7	r3	r3	r3	r3	r3		
8	r1	r1	r1	r1	r1		

Check whether the below mentioned strings can be parsed by this parser. [2]
i) id + (id + id)
ii) (id + id + id)

(a) i) Accept, ii) Error

(b) i) Error, ii) Error

(c) i) Accept, ii) Accept

(d) i) Error, ii) Accept

Answer: c)