Compiler
2018 Fall Middle Examination

Name_____ Student No._____ Score_____

Problem 1: (40 points)

1

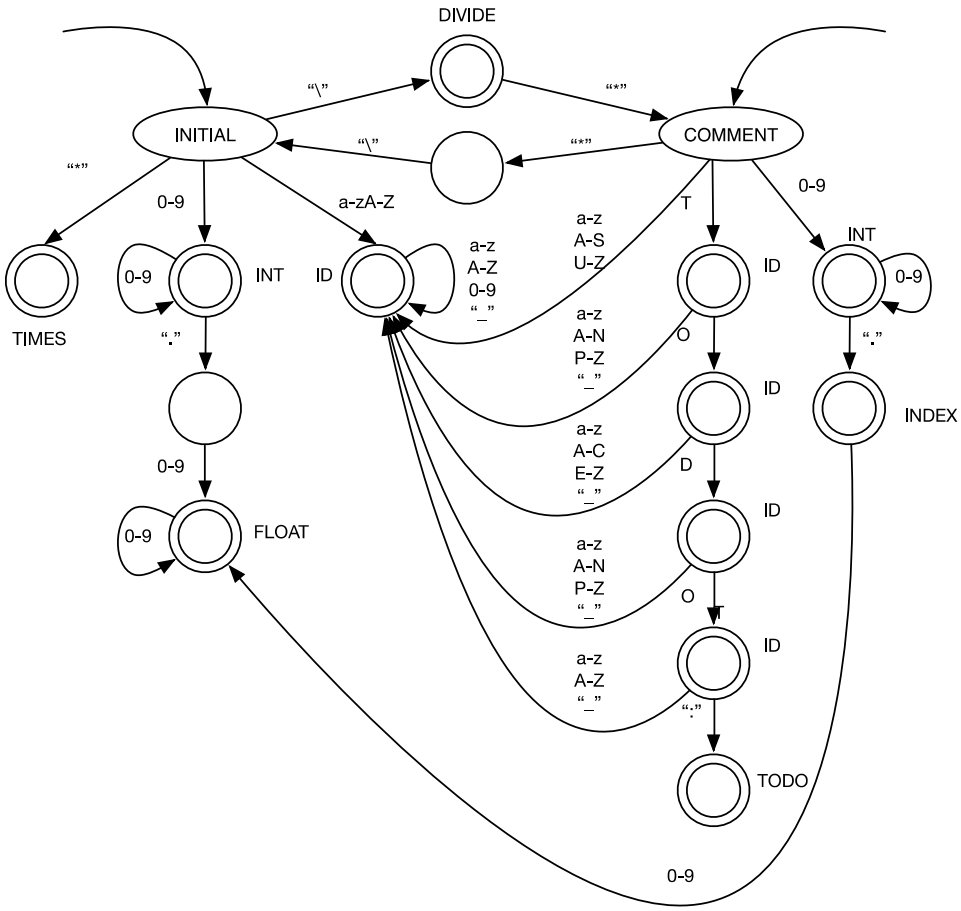
（1.1） **id = [a-zA-Z][a-zA-Z0-9"_"]***

（1.2） **int = [0-9]+**

（1.3） **float = int"."int**

（1.4） **index = int"."**

2



3

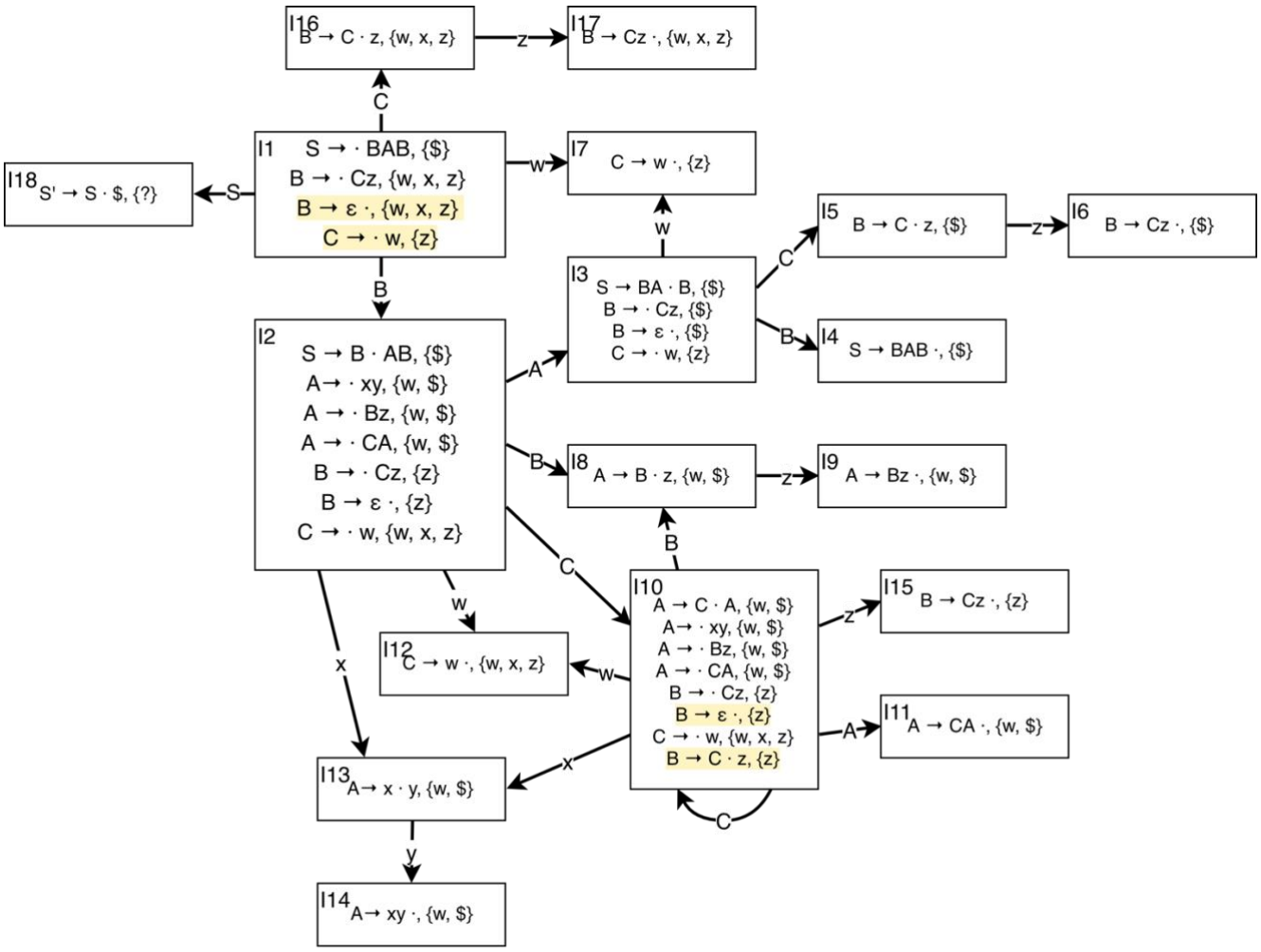
a96669666690513142

Problem 2: (60 points)

1

Terminal(s)	w, x, y, z, $
Non-terminal(s)	S, A, B, C
First Set(s)	First(S) = {w, x, z}; First(A) = {w, x, z}; First(B) = {w}; First(C) = {w};

2



3

	$	w	x	y	z	S	A	B	C
1		S7, R6	R6		R6	G18		G2	G16
2		S12	S13		R6		G3	G8	G10
3		S7						G4	G5
4	R1								
5					S6				
6	R5								
7					R7				
8					S9				
9	R3	R3							
10		S12	S13		S15, R6		G11	G8	G10
11	R4	R4							
12		R7	R7		R7				
13				S14					
14	R2	R2							
15					R5				
16		S17	S17		S17				
17		R5	R5		R5				
18	Accept								
19									
20									
21									
22									
23									
24									
25									
26									
27									
28									

4

wzzwz$	reduce first		wzzwz$	shift first
stack	action		stack	action
1B2	R6		1w7	shift
1B2w12	shift		1C16	R7
1B2C16	R7		1C16z17	shift
1B2C16z17	shift		1B2	R5
1B2B8	R5		1B2B8	R6
1B2B8z9	shift		1B2B8z9	shift
1B2A3	R3		1B2A3	R3
1B2A3w7	shift		1B2A3w7	shift
1B2A3C5	R7		1B2A3C5	R7
1B2A3C5z6	shift		1B2A3C5z6	shift
1B2A3B4	R5		1B2A3B4	R5
1S18	R1		1S18	R1
	Accept			Accept

Problem 1: Lexical Analysis (40 points)

```
/* lex definition */

id                                                    (1.1)

int                                                   (1.2)

float                                                 (1.3)

index                                                 (1.4)


/* start states*/

%Start INITIAL COMMENT


%%

/* regular expressions and actions */

<INITIAL>id                          {ADJ; printf("0"); return ID;}

<INITIAL>int                         {ADJ; printf("1"); return INT;}

<INITIAL>float                       {ADJ; printf("2"); return FLOAT;}

<INITIAL>"*"                         {ADJ; printf("3"); return TIMES;}

<INITIAL>"/"                         {ADJ; printf("4"); return DIVIDE;}

<INITIAL>"="                         {ADJ; printf("5"); return EQUAL;}

<INITIAL>"/*"                        {ADJ; BEGIN(COMMENT);}

<COMMENT>id                          {ADJ; printf("6"); return ID;}

<COMMENT>int                         {ADJ; printf("7"); return INT;}

<COMMENT>float                       {ADJ; printf("8"); return FLOAT;}

<COMMENT>index                       {ADJ; printf("9"); return INDEX;}

<COMMENT>"TODO:"                     {ADJ; printf("a"); return TODO;}

<COMMENT>"*/"                        {ADJ; BEGIN(INITIAL);}

.                                    {ADJ; return SPACE;}
```

In this problem, let's consider an interesting lexical analyzer, which is able to analyze the contents of a comment, instead of simply ignoring them. With the related lex code listed above, please answer the following questions.

1. Regular Expression (8')

   At first, you need to write down the regular expressions of the identifiers(id), integers(int), float numbers(float), and indexes(index) (1.1~1.4). Their descriptions are as follow:

   (a) An identifier is composed of any characters from lower and upper case English letters, digits, and "_" (underline), but only a letter is allowed as the first character of an identifier.

   (b) An integer is composed of digits only.

   (c) A float number consists of an integer part and a fractional part linked by a "." (dot), both parts contain digits only and **cannot** be empty.

   (d) An index is an integer followed by a "." (dot).

Valid examples:

compilers (id) 666 (int) 666.6 (float) 6. (index)

Six_compilers (id) 000 (int) 000.00(float) 0. (index)

Invalid examples:

6_compilers (digit beginning)

.666 (empty integer part)

2. Deterministic Finite Automaton (DFA) (30')

   Draw the minimized DFA on the following requirements. For any regular expression, the minimized DFA is a unique DFA having the smallest number of states that accepts it. You will get part of scores if your DFA is not minimized. And your DFA need accept every state in the lex code listed above **except** the SPACE state.

3. What will be the output on the following input? Assume there is nothing to the right of the last visible character on each line except a newline character and that the following input is to be scanned all at once: (2')

```
/*
TODO:
1. Finish this exam.
2. Start to do lab 4.
*/
score = 37 * 36 / 20.0
```

Problem 2: Grammar (60 points)

Let ***G*** be the grammar:

0)	S′ → S$
1)	S → BAB
2)	A → xy
3)	A → Bz
4)	A → CA
5)	B → Cz
6)	B → ε
7)	C → w

Note: the "ε" means "B" can derive the empty string. (nullable(B) == true)

Using the above grammar ***G***, answer the following questions.

1. What are the terminals and non-terminals of this grammar? Give the First sets for each non-terminal in the grammar.

2. Construct the state graph (DFA) for this grammar using LR(1) parsing.

3. Follow the LR(1) procedure to construct the parsing table for the above DFA. Does there exist any conflict? If so, please list out the conflict(s) and the type(s) of the conflict(s). If not, please write "**No conflict**".

4. Given the input string "wzzwz", show the procedure of the stack and the corresponding actions using the parser above. (Note: If there exist any conflict(s), you should firstly specify the priorities of ***shift*** and ***reduce*** operations as you like before parsing)