



มหาวิทยาลัยสงขลานครินทร์  
คณะวิศวกรรมศาสตร์

การสอบปลายภาค ประจำปีการศึกษาที่ : 1

ประจำปีการศึกษา: 2547

วันที่: 7 October 2547

เวลา: 13:30-16:30

วิชา: 240-340 Compiler Structures

ห้อง: 1205/ หัวหุ่น

ทฤษฎีในการสอบ โทษขั้นต่ำคือ ปรับตกในรายวิชาที่ทฤษฎี และพักการเรียน 1 ภาคการศึกษา

คำสั่ง: อ่านรายละเอียดของข้อสอบ และคำแนะนำให้เข้าใจก่อนเริ่มทำข้อสอบ

อนุญาต: เครื่องเขียนต่างๆ เช่น ปากกา หรือดินสอ

ไม่อนุญาต: หนังสือ, เอกสารใดๆ และเครื่องคิดเลข

เวลา: 3 ชั่วโมง (180 นาที)

ชื่อ \_\_\_\_\_ รหัสนักศึกษา \_\_\_\_\_ Section \_\_\_\_\_

Notational conventions:

- non-terminal symbols of a grammar are in boldface Courier lower-case letters; example: `term`
- terminal symbols (tokens) are in boldface Courier upper-case letters, and are either given literally in single quotes, or as a token name; examples: `' ! ' NUMBER ' <= '`
- the symbols `→` and `|` mark off the parts of a production
- the symbol `ε` represents the null or empty string of zero characters

1. The grammar

$$e \rightarrow e \text{ '+' } e \mid e \text{ '*' } e \mid \text{NUMBER}$$

has the operator-precedence relations as in the following table:

	NUMBER	'+'	'*'	\$
NUMBER		•>	•>	•>
'+'	<•	•>	<•	•>
'*'	<•	•>	•>	•>
\$	<•	<•	<•	

The input to be parsed is:

$$\$3+4*7*2\$$$

a) Show the string with the precedence relations inserted.

b) Show the parsing, step by step, including the productions reduced by, and the parse tree after each reduction.

## 2. The grammar

bexpr → bexpr OR bterm  
 | bterm

bterm → bterm AND bfactor  
 | bfactor

bfactor → NOT bfactor  
 | '(' bexpr ')'  
 | TRUE  
 | FALSE

has the parse table:

state	Action								goto		
	AND	OR	NOT	TRUE	FALSE	(	)	\$	bexpr	bterm	bfactor
0			s2	s3	s1	s4			5	6	7
1	r8	r8	r8	r8	r8	r8	r8	r8			
2			s2	s3	s1	s4					8
3	r7	r7	r7	r7	r7	r7	r7	r7			
4			s2	s3	s1	s4			9	6	7
5		s11						s10			
6	s12	r2	r2	r2	r2	r2	r2	r2			
7	r4	r4	r4	r4	r4	r4	r4	r4			
8	r5	r5	r5	r5	r5	r5	r5	r5			
9		s11					s13				
10								accept			
11			s2	s3	s1	s4				14	7
12			s2	s3	s1	s4					15
13	r6	r6	r6	r6	r6	r6	r6	r6			
14	s12	r1	r1	r1	r1	r1	r1	r1			
15	r3	r3	r3	r3	r3	r3	r3	r3			

Show the progress of the parse, the rules reduced by, and changes to the parse tree for the input

\$ NOT ( FALSE OR TRUE ) AND TRUE \$

3. Suppose that the grammar:

```
bexpr → bexpr '||' bterm
      | bterm
bterm → bterm '&&' bfactor
      | bfactor
bfactor → '!' bfactor
        | '(' bexpr ')'
        | 'true'
        | 'false'
```

is to be programmed into an interpreter.

a) Show the rules (regular expressions and actions) that would be needed in the second part of a `lex` or `flex` input file. Use capital-letter token names that begin with `"TOKEN_"`.

b) Assume that the file `"boolean_types.h"` contains the declarations:

```
typedef int bool_t;
#define false ((bool_t)0)
#define true ((bool_t)1)
```

Show the additional declarations in this header file for structures and enumerations to store the data that `bexpr`, `bterm` and `bfactor` nodes need to remember in a parse tree.

C) Show the method prototypes that could be placed in a "boolean\_methods.h" header file for 6 methods that could be called to create and evaluate the bexpr, bterm and bfactor tree nodes above.

d) How would you tell `bison` what the goal or start symbol of this grammar was? In which section of the parser description file does it go?

e) Show the `bison` declaration to say that the parser value stack can contain pointers to the `bexpr`, `bterm` and `bfactor` node types, as well as Boolean constants for the `TRUE` and `FALSE` tokens? In which section of the input file does this go?

f) Show parser input to name the tokens (in capital letters, beginning with “`TOKEN_`”), declare the associativity and precedence of the operator tokens (logical `OR` ‘`||`’ is less than logical `AND` ‘`&&`’ which is less than the unary `NOT` ‘`!`’ operator; `OR` and `AND` are left associative), and set the type of node pointer that the non-terminal symbols of the grammar put on the parse value stack. In which section of the input file does this go?

g) If the parser is described in a file named "boolean.y", and "bison -d -t -v" produces the default output files, what text must be included in the first part of the `lex` or `flex` input file to declare the token values the same as in the parser, and make the node structure types and the `create` and `evaluate` method prototypes known to the parser?

h) Show the rules (productions and actions) that would be placed in the parser description input file. Use lower case non-terminal names, as suggested in the web page article.

i) Show the code for the method to create a bexpr node.

j) Show the code for the method to evaluate a bexpr node.



K) Show the code for the method to create a bfactor node.

L) Show the code for the method to evaluate a bfactor node.