cmps104a-2011q4-exam3.mm:496: warning: 'V+' not defined

page 1 page 2 page 3 page 4 page 5 Total / 52

Please print clearly:

Name: ANSWERS

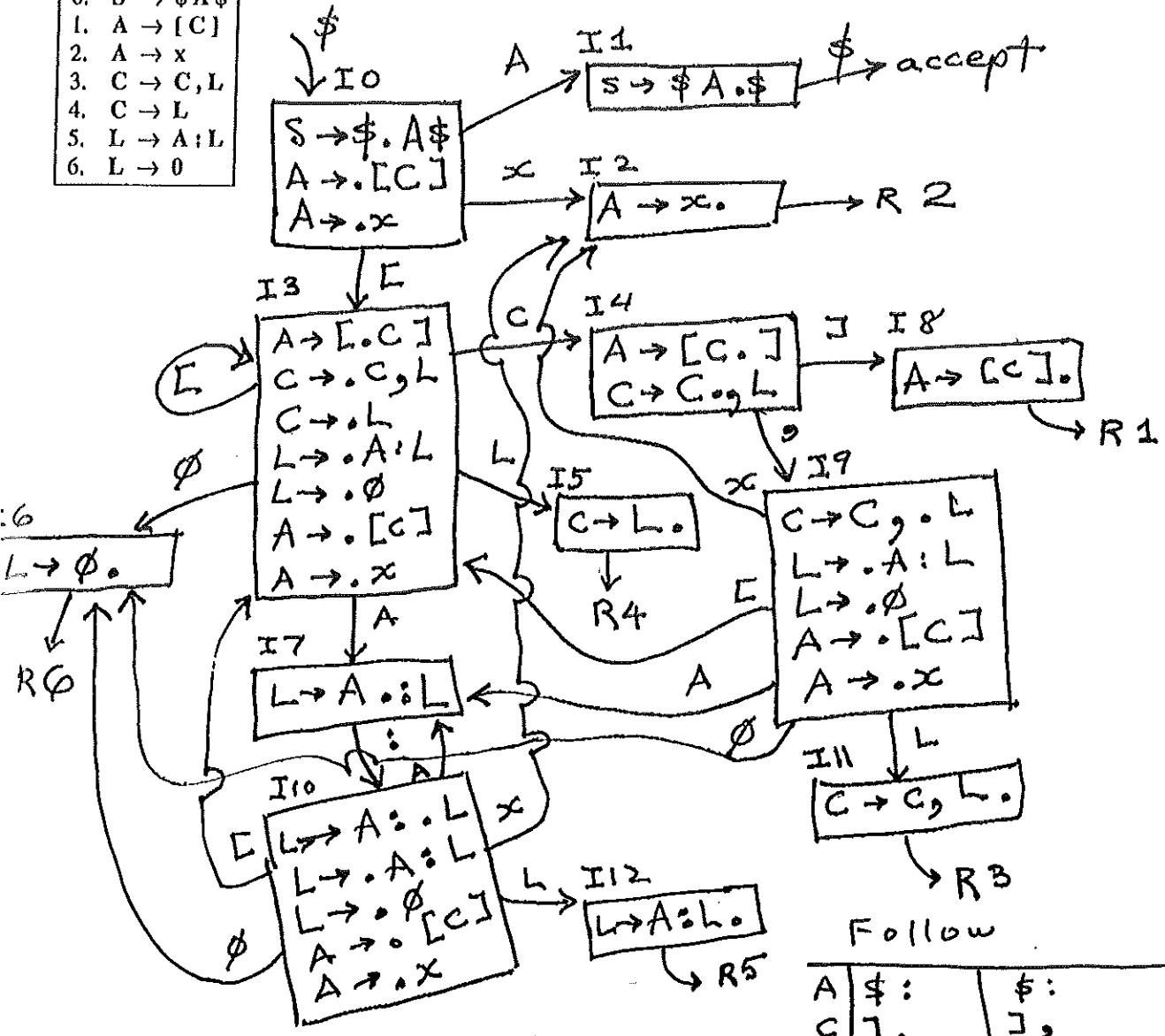
Login: @ucsc.edu

No books; No calculator; No computer; No email; No internet; No notes; No phone. Neatness counts! Do your scratch work elsewhere and enter only your final answer into the spaces provided.

1. Given the grammar presented here, and using the style from the LALR(1) handout:
   (a) Construct the characteristic finite state machine (CFSM), sets of items and transition diagram, showing shifts, reductions, and acceptance. [6✓]
   (b) Construct the FOLLOW sets. [3✓]
   (c) Answer *yes* or *no* to each of the following questions: [1✓]

   Is the grammar LR(0)? yes Is the grammar SLR(1)? yes

0. S → A
1. A → [C]
2. A → x
3. C → C,L
4. C → L
5. L → A:L
6. L → ∅



	Follow	
A	$:	$:
C	],	],
L	R4 R3 R5	],

8 × 1
1 × 2

2. Define a grammar using `bison`, showing everything that needs to be placed into section 1 and 2 of the grammar. Do not show any semantic actions. Make your grammar as brief as possible by leveraging on ambiguity declarations in section 1. The language is defined below. [5✓]
   (a) A program is a sequence of one or more expressions separated by commas. Commas may only occur between expressions, never at the beginning nor end of the program.
   (b) An expression is one of the following:
      (i) An `if` keyword followed by an expression follwed by a `then` keyword followed by an expression. Optionally, this may be followed by an `else` keyword followed by another expression.
      (ii) Two expressions connected by a plus (+) sign, left associative and a precedence above that of `if`.
      (iii) An identifier followed by a subscript.
      (iv) An expression inside parentheses ((and)).
      (v) Either an identifier or a number.
   (c) A subscript is an expression inside brackets ([and]).

```
%token IF THEN ID NUM
%right ELSE
%left '+'
%start P
%%
P : P ',' e
  | e
  ;
e : IF e THEN e ELSE e
  | IF e THEN e %prec ELSE
  | e '+' e
  | ID '[' e ']'
  | '(' e ')'
  | ID | NUM ;
```

3. Define the lexical grammar for the language above, such that it can be compiled using `flex`. Forsemantic actions, just show a `return` statement where a token is returned. Define macros in section 1 and use them in section 2, as appropriate. [5✓]
   (a) An identifier is one or more upper- or lower-case letters, digits, or underscores, but may not begin with a digit.
   (b) A number is a sequence of one or more digits with an optional decimal point. The decimal point may occur in front of or after all digits, or between a pair of digits.
   (c) Comments are two slashes, as in Java, followed by all characters up to, but not including, the next newline character. (The C preprocessor is not used.)
   (d) White space is any sequence of one or more spaces, tabs, or newline characters.
   (e) Other tokens are as described in the `bison` grammar above.

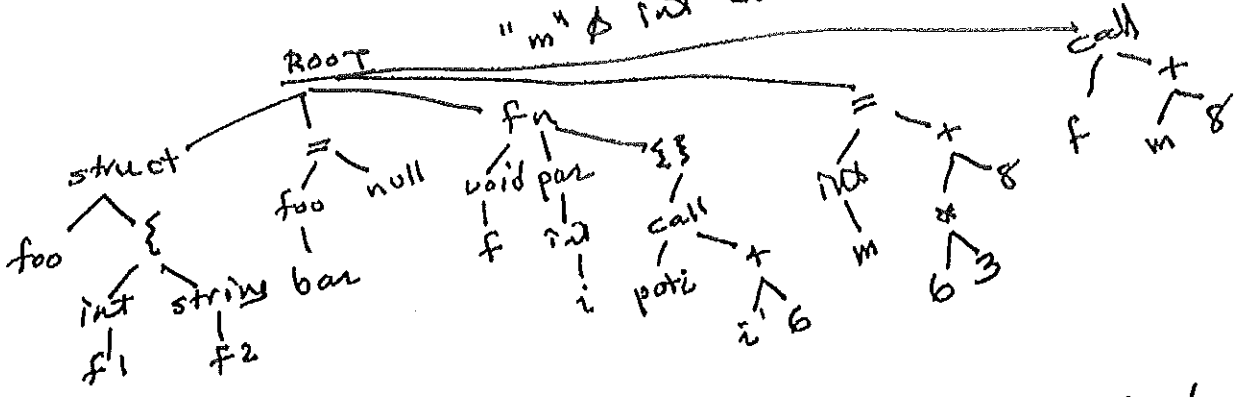
```
DIG   [0-9]
LET   [A-Za-z_]
%%
{LET}({LET}|{DIG})*        {return ID;}
{DIG}+\.?{DIG}*|\.{DIG}+   {return NUM;}
"//".*
[ \t\n]+
"if"                       {return IF;}
"then"                     {return THEN;}
"else"                     {return ELSE;}

"+"   {return '+';}
"["   {return '[';}
"]"   {return ']';}
"("   {return '(';}
")"   {return ')';}
","   {return ',';}
```

4. Given the program at the left:
   (a) Show the symbol table at the right, specifying which symbol table each has each identifier. Use the specification for project 4. Do not show any hashing structure, just list the fields, block numbers, and attributes as appropriate. The attributes to choose from are: **void, bool, char, int, null, string, struct, array, function, variable, field, typeid, param, lvalue, const, vreg, vaddr.** [3✓]
   (b) Draw the abstract symtax tree as per the project 3 specifications (Don't show attributes). [3✓]
   (c) Translate this program into oil code as per the project 5 specifications. [4✓]

```
struct foo {
   int f1;
   string f2;
}
foo bar = null;
void f (int i) {
   puti (i + 6);
}
int m = 6 * 3 + 8;
f(m + 8);
```

struct
∅ "foo" struct typeid → field
"f1" int field ∅
"f2" string field ∅

ident
"bar" struct lval var ∅
"f" void function ∅
"i" int variable param lvalue 1
↳ ∅
"m" ∅ int var lvalue




```
struct _foo {
   int _f1;
   char *_f2;
};
struct foo * bar;
void f(
   int i;
){
   int t1 = i+6;
   puti(t1)
}
int m;
```

```
void _oc_main (void) {
   bar = ∅;
   int t2 = 6 * 3
   int t3 = t2+8;
   m = t3;
   int t4 = m+8;
   f(t4)
}
```

Multiple choice. To the *left* of each question, write the letter that indicates your answer. Write Z if you don't want to risk a wrong answer. Wrong answers are worth negative points. [11✓]

number of correct answers		× 1 =	$= a$
number of wrong answers		× ½ =	$= b$
number of missing answers		× 0 =	0
column total $c = \max(a - b, 0)$	11		$= c$

1. Which of the following items in a state will cause a reduction?
   (A) $E \rightarrow \bullet\ E + T$
   (B) $E \rightarrow E \bullet + T$
   (C) $E \rightarrow E + \bullet\ T$
   (D) $E \rightarrow E + T \bullet$

   D

2. Which of the following items was entered into a state during the closure operation?
   (A) $E \rightarrow \bullet\ E + T$
   (B) $E \rightarrow E \bullet + T$
   (C) $E \rightarrow E + \bullet\ T$
   (D) $E \rightarrow E + T \bullet$

   A

3. Which item will cause a shift action to be added to the state during goto propagation?
   (A) $E \rightarrow \bullet\ E + T$
   (B) $E \rightarrow E \bullet + T$
   (C) $E \rightarrow E + \bullet\ T$
   (D) $E \rightarrow E + T \bullet$

   B

4. What might one expect as the output from compiling and running the following C program?
   `void main() { printf ("%p\n", main); }`
   (A) `0x4004c8`
   (B) `Segmentation fault`
   (C) `UHÅ¿È@`
   (D) `a.out: Command not found.`

   A

5. In general, a DFA is _____ when compared to an NFA.
   (A) faster (CPU) and larger (memory)
   (B) faster (CPU) and smaller (memory)
   (C) slower (CPU) and larger (memory)
   (D) slower (CPU) and smaller (memory)

   A

6. A reduce/reduce conflict occurs in an LR(1) machine when:
   (A) The grammar is ambiguous, and only when the grammar is ambiguous.
   (B) A terminal symbol is in the lookahead set of a reduction, and in the lookahead set of another reduction in the same state.
   (C) A terminal symbol is in the lookahead set of a reduction, and also on an outgoing transition in the same state.
   (D) A nonterminal symbol is in the lookahead set of a reduction, and also on an outgoing transition in the same state.

   B

7. An input file containing n characters will be scanned by a DFA in what time?
   (A) $O(\log_2 n)$
   (B) $O(n)$
   (C) $O(n \log_2 n)$
   (D) $O(n^2)$

   B

8. A sequence of straight-line code fundamental to code optimization, such that if the first instruction is executed, then so will all of the following instructions be executed, describes a:
   (A) basic block
   (B) folded constant
   (C) natural loop
   (D) stack machine

   A

9. Which language can be recognized by a push-down αὐτόματον but not by a finite state machine?
   (A) A sequence of one or more decimal digits.
   (B) An x followed by zero or more y's.
   (C) Input requiring balanced parentheses.
   (D) Strictly alternating x's and y's. Example: xyxyxyxyxy

   C

10. A `flex` pattern that matches either an x or a y, but not both:
    (A) `x*y`
    (B) `x+y`
    (C) `x?y`
    (D) `x|y`

    D

11. What is it called when the lookahead symbol is pushed onto the parsing stack, then the scanner is called to obtain the next lookahead symbol?
    (A) accept
    (B) error
    (C) reduce
    (D) shift

    D

Multiple choice. To the *left* of each question, write the letter that indicates your answer. Write *Z* if you don't want to risk a wrong answer. Wrong answers are worth negative points. [11✓]

number of correct answers		× 1 =	$= a$
number of wrong answers		× ½ =	$= b$
number of missing answers		× 0 =	0
column total $c = \max(a - b, 0)$	11		$= c$

1. What is the appropriate semantic action to attach to the `bison` rule
   `expr : '(' expr ')'`
   (A) `{ $$ = $1; }`
   (B) `{ $$ = $2; }`
   (C) `{ $1 = $$; }`
   (D) `{ $2 = $1; }`

2. In a local stack frame, the static (access) link points at:
   (A) The instruction following the call instruction that activated the function.
   (B) The local stack frame in which the current function is nested.
   (C) The local stack frame of the caller.
   (D) The object referred to as `this` in Java and C++.

3. During a function call, the return address is pushed onto the stack or placed in a register by the:
   (A) called function
   (B) calling function
   (C) garbage collector
   (D) operating system

4. Given `String s` and `Object o` in Java, if `o` actually points an an `Integer`, when will the error in the following statement be detected?
   `s = (String) o;`
   (A) by the parser
   (B) by the type checker
   (C) by the code generator
   (D) by the runtime routines

5. For a grammar $G = \langle V_N, V_T, P, S \rangle$, where the elements of P are of the form $(A \rightarrow \beta)$, where:
   (A) $\beta \in (V_N \cap V_T)^*$
   (B) $\beta \in (V_N \cup V_T)^*$
   (C) $\beta \in V_N^*$
   (D) $\beta \in V_T^*$

6. A sequence of instructions such that, if the first one in the sequence is executed, so will all of the others, barring an exception, is called a:
   (A) basic block
   (B) function
   (C) interpreter
   (D) natural loop

7. A form of automatic memory management which fails to work on a cyclic data structure is:
   (A) copying collector with semispaces
   (B) explicit allocation and deallocation
   (C) mark and sweep
   (D) reference counting collector

8. The Java virtual machine (JVM) is an interpreter which uses:
   (A) abstract syntax tree
   (B) stack machine code
   (C) three address code
   (D) two address code

9. An ambiguous grammar like the following can be disambiguated in `bison` by declaring `else` to be __(a)__ associative with a very low precedence so that preference will be given to a __(b)__ when is the lookahead symbol. Assume a C-like language.
   `stmt : IF '(' expr ')' stmt ELSE stmt`
   `stmt : IF '(' expr ')' stmt %prec ELSE`
   (A) (a) = left, (b) = reduce
   (B) (a) = left, (b) = shift
   (C) (a) = right, (b) = reduce
   (D) (a) = right, (b) = shift

10. If we represent a set of the integers 0 to 31 as a `uint32_t` bitset, the union of two such sets can be expressed in C as:
    (A) `x | y`
    (B) `x & y`
    (C) `x ^ y`
    (D) `x ~ y`

11. "Go To Statement Considered Harmful"
    (A) Edsger Dijkstra
    (B) Donald Knuth
    (C) John McCarthy
    (D) Dennis Ritchie