`$Id: cmps104a-2016q2-exam2.mm,v 1.63 2016-05-09 14:38:56-07 - - $`

page 1	page 2	page 3	Total / 32

Please print clearly :

Name : SOLUTION	
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. Points will be deducted for messy or unreadable answers.

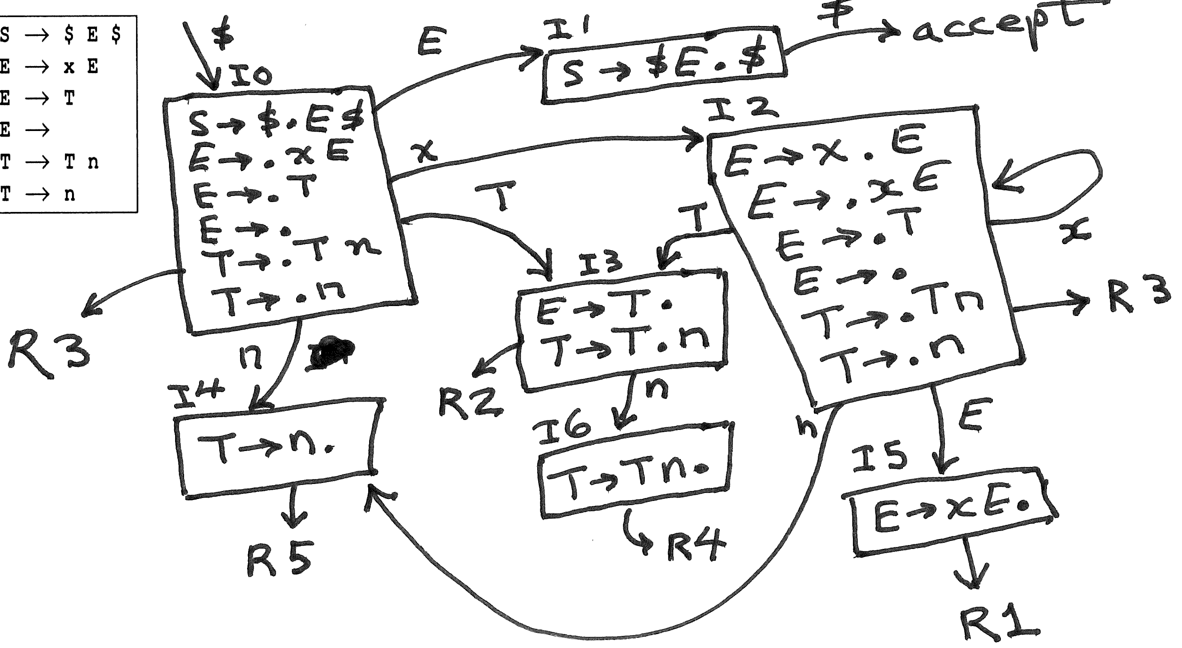
1. Code a simplified version of the function `malloc`. Assume that the free area of the heap is contiguous with the pointer `begin` pointing at its first byte and the pointer `end` pointing at the end of the heap. Maintain the invariant that `begin` is a multiple of 16. Use `assert` to crash the program if allocation reaches beyond `end`. **[2✓]**

```
void* malloc (size_t nbytes) {
        nbytes = (nbytes + 15) & ~15;
        void* t = begin;
        begin += nbytes;
        assert (begin <= end);
        return t;
```

2. 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 shift, reduce, and accept actions. **[5✓]**
   (b) Construct the Follow sets. Show the first pass with rule symbols in the Follow sets. Then show the revised follow sets with only terminal symbols. (Use the charts at the bottom of the page.) **[2✓]**
   (c) Answer *yes* or *no* to each of the following questions : **[1✓]**

   Is the grammar LR(0) ? NO Is the grammar SLR(1) ? NO

0.	S → $ E $
1.	E → x E
2.	E → T
3.	E →
4.	T → T n
5.	T → n



Follow sets with rule symbols:

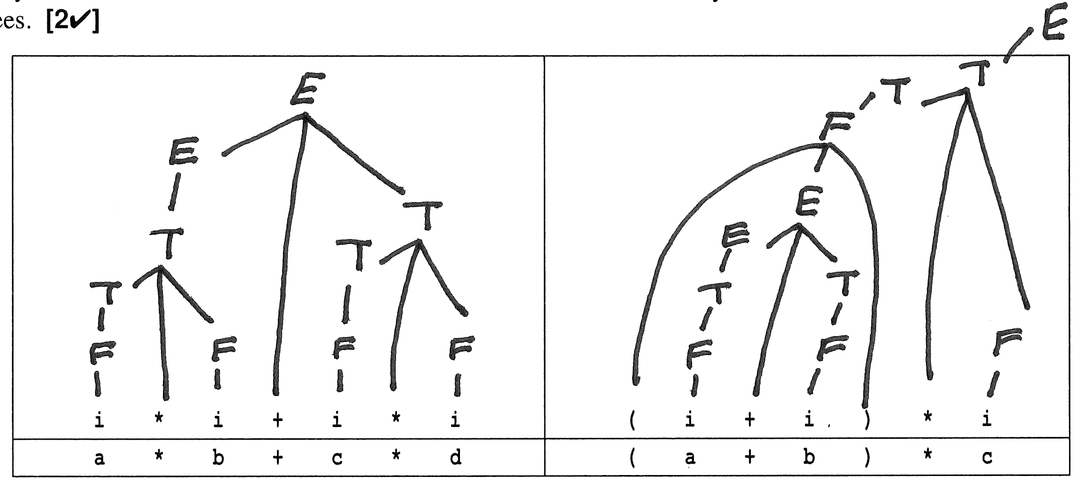
Follow(E) : $ R₁	
Follow(T) : n R₂	

Follow sets with only V_T symbols :

Follow(E) : $	
Follow(T) : n $	

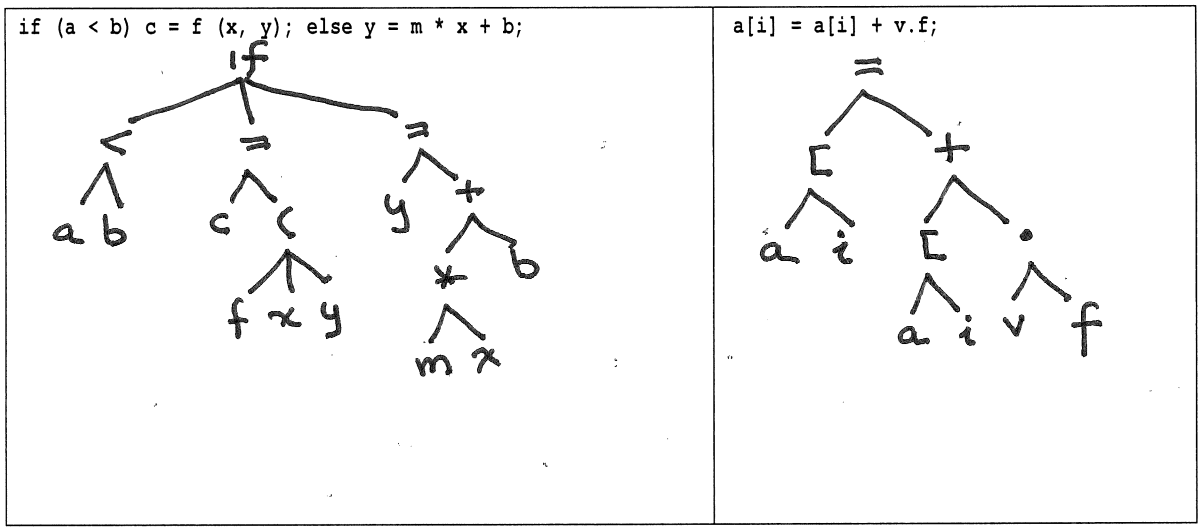
3. Draw parse trees (not abstract syntax trees) for the following expressions, given the ETF grammar at the left. The lexical symbols are below the middle line. Match each with a terminal symbol above the middle line and draw the trees. **[2✓]**

$E \rightarrow E + T$
$E \rightarrow T$
$T \rightarrow T * F$
$T \rightarrow F$
$F \rightarrow (E)$
$F \rightarrow i$



4. A context-free grammar $G = \langle V_N, V_T, P, S \rangle$. If P is the ETF grammar above, fill in the blanks for the other three components. **[1✓]**
$V_N = \{$ E T F $\}$, $V_T = \{$ + * ( ) i $\}$, $S =$ E

5. Draw abstract syntax trees (not parse trees) according to the project 3 specifications for each of the following statements. **[3✓]**



```
if (a < b) c = f (x, y); else y = m * x + b;
```

```
a[i] = a[i] + v.f;
```

6. Define a grammar using **bison** for a language described as follows. Show only the syntax rules. Do not show any semantic actions. Clearly indicate what goes above the first `%%` and what goes below. **[4✓]**
   (a) A program consists of zero or more statements, each terminated with a semi-colon.
   (b) A statement is a function call.
   (c) A function call is an identifier, followed by a left parenthesis, followed by zero or more arguments separated by commas, followed by a right parenthesis.
   (d) An argument is a function call, an identifier, or a number.

```
%token ID NUM
%start prog
%%
prog : prog stmt | ;
stmt : call ';' ;
call : ID '(' ')'
     | ID '(' args ')'
     ;
args : args ',' arg
     | arg
     ;
arg : call | ID | NUM ;
```

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. **[12✓]**

number of correct answers		× 1 =	$= a$
number of wrong answers		× ½ =	$= b$
number of missing answers		× 0 =	0
column total $c = \max(a - b, 0)$	12		$= c$

D 1. For the x86_64 architecture, how many registers are there, and how many bits are there in each register ?
(A) 8 registers, 32 bits
(B) 8 registers, 64 bits
(C) 16 registers, 32 bits
(D) 16 registers, 64 bits

A 2. Which grammar is ambiguous ?
(A) $E \to E + E$
$T \to x$
(B) $E \to E + T$
$T \to x$
(C) $E \to T + E$
$T \to x$
(D) $E \to T + T$
$T \to x$

C 3. Which grammar makes the `+` operator right associative and allows it to occur an arbitrary number of times ?
(A) $E \to E + E$
$T \to x$
(B) $E \to E + T$
$T \to x$
(C) $E \to T + E$
$T \to x$
(D) $E \to T + T$
$T \to x$

A 4. Which might be a function prolog ?
(A) `push bp; bp = sp; sp -= N;`
(B) `push sp; sp = bp; bp -= N;`
(C) `sp -= N; push bp, sp = bp;`
(D) `sp = bp; push sp; sp -= N;`

D 5. Which might be a function epilog ?
(A) `bp = sp; pop sp; return;`
(B) `pop bp; sp = bp; return;`
(C) `pop sp; bp = sp; return;`
(D) `sp = bp; pop bp; return;`

C 6. Memory recycling using reference counting fails for which of the following data structures ?
(A) acyclic data structure
(B) binary search tree
(C) cyclic data structure
(D) linear linked list

C 7. A mark and sweep garbage collector marks all object of what class in order to keep them from being recycled ?
(A) dead
(B) live
(C) reachable
(D) unreachable

B 8. Calls to `new` or `malloc` will allocate memory on which program segment ?
(A) data
(B) heap
(C) stack
(D) text

C 9. Which `bison` declarations would be appropriate in a language with the same precedence and associativity as C/C++/Java ?
(A) `%left '*' '/'`
`%left '+' '-'`
(B) `%right '*' '/'`
`%right '+' '-'`
(C) `%left '+' '-'`
`%left '*' '/'`
(D) `%right '+' '-'`
`%right '*' '/'`

A 10. If `malloc` returns successfully, what might be printed by the following statement ?
`printf ("%p\n", malloc(256));`
(A) `0x10cb030`
(B) `0x10cb037`
(C) `0xabcdefg`
(D) `012345678`

B 11. When is a static variable is assigned a specific virtual address ?
(A) compile time
(B) link time
(C) when a function is called
(D) when the program is loaded

A 12. Which is true ?
(A) All LR(k) languages are unambiguous.
(B) All unambiguous languages are LR(k).
(C) Some LR(0) languages are not SLR(1).
(D) Some LR(k) languages are ambiguous.