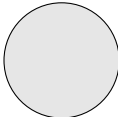
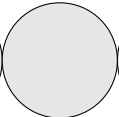
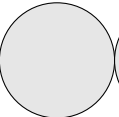
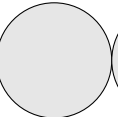
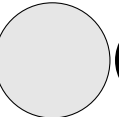
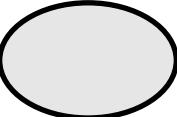


page 1	page 2	page 3	page 4	page 5	Total / 50	Please PRINT using keyboard letters : Name : Login : @ucsc.edu
						

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) ? _____ Is the grammar SLR(1) ? _____

- | | |
|----|-------------------------|
| 0. | $S \rightarrow \$ E \$$ |
| 1. | $E \rightarrow E + T$ |
| 2. | $E \rightarrow T$ |
| 3. | $T \rightarrow * T$ |
| 4. | $T \rightarrow x$ |

2. Examine the program listed here on the left. Note that the bodies of two functions have been omitted. When name mangling involves block numbers, use the sequence 1, 2, 3, etc. Fill in the symbol table entries for all identifiers, showing their mangled names. Show all applicable attributes from the following set: **int**, **char**, **void**, **array[n]**, **ptr(n)**, **function**, **lval**, **global**, **local**, **param**. For pointers, n is the number of levels. For arrays, n is the dimension. **[4✓]**

line	// code	identifier	attributes
	int smallest (int *a,		
	int i, int n);		
	void swap (int *x, int *y);		
1.	void selectionsort (
2.	int *a,		
3.	int n) {		
4.	int i = 1;		
5.	while (i < n) {		
6.	int min = smallest		
7.	(a, i, n);		
8.	swap (&a[i - 1],		
9.	&a[min]);		
10.	i = i + 1;		
11.	}		
12.	}		

3. Translate the function **selectionsort** into icode, as per project 5. Enter each line of code into the box whose number corresponds to the numbered lines of code above. Feel free to optimize the code. Do not declare any **tmp** variables. Just number the temps as **t1**, **t2**, etc. Translate only lines 1 through 12. **[4✓]**

1.	7.
2.	8.
3.	9.
4.	10.
5.	11.
6.	12.

4. Draw a flow graph of the function **selectionsort** (supra), showing a circle for each basic block. Inside each circle write the numbers 1 to 12, showing which statements belong to each basic block. Draw the dominator tree for your flow graph. Do not show the depth-first spanning tree. **[2✓]**

5. Write the code for **malloc**, assuming a copying collector with semispaces. All of the allocated data, therefore, is packed into low addresses, and there is one huge free area, with the pointer **free** pointing at the lowest address in the free area, and the pointer **end** pointing at the first byte beyond the free area. Your code should call **gc**, the garbage collector if necessary. [2✓]

6. Using Thompson's construction, draw the non-deterministic finite αὐτόματον derived from the **flex** regular expression shown below this paragraph. Then for each state, make a table of the ε̂ ψιλόν closures. From that draw the derived deterministic finite αὐτόματον. [3✓]

ab|cd

7. Using **bison** syntax, define the grammar outlined here. Your grammar must be written unambiguously, and be acceptable input to **bison**. [3✓]
- (a) The start symbol is **expr**, which consists of a sequence of **terms** connected together via one of the ***** or **+** operators, which are right associative.
 - (b) A **term** is a sequence of **factors** connected by the **/** operator, which is left associative.
 - (c) A **factor** is a parenthesized **expr**, or an identifier.

8. Write the following patterns using **flex** notation : [2✓]
- (a) A string constant which begins and ends with a double quote mark (**"**). Between the quote marks are zero or more character elements, none of which may be a backslash (****), double quote, or a newline. However, escaped characters are recognized within the string, namely ****, **\"**, and **\.**
 - (b) A floating point literal, which consists of a fraction and an optional exponent. The fraction consists of a sequence of one or more digits, with a single decimal point at the front, at the end, between a pair of digits, or omitted. The exponent is an upper- or lower-case letter **E**, following by an optional sign, followed by one or more digits.

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		$\times 1 =$	$= a$
number of wrong answers		$\times \frac{1}{2} =$	$= b$
number of missing answers		$\times 0 =$	0
column total $c = \max(a - b, 0)$	11		$= c$

- What kind of garbage collection is unable to handle cyclic data structures ?
(A) copying collector with semispaces
(B) generational
(C) mark and sweep
(D) reference counting
- Which of the following is *not* used to identify the leader of a basic block ?
(A) entry instruction of a function
(B) instruction following a jump
(C) target of a jump instruction
(D) target of a return instruction
- Given a regular expression r , of length $|r|$, the NFA constructed from it will have at most (a) states, and when used to scan a string s , of length $|s|$, will take (b) time.
(A) (a) = $O(2^{|r|})$; (b) = $O(|r| \times |s|)$.
(B) (a) = $O(2^{|r|})$; (b) = $O(|s|)$.
(C) (a) = $O(|r|)$; (b) = $O(|r| \times |s|)$.
(D) (a) = $O(|r|)$; (b) = $O(|s|)$.
- Thompson's construction :
(A) converts a DFA into an NFA
(B) converts a regex into an NFA
(C) converts an NFA into a DFA
(D) minimizes a DFA
- What is used to identify the natural loops of a function ?
(A) basic block
(B) dominator tree
(C) flow graph
(D) tail call
- Given the following statement,
`for (i = 0; i < n; ++i) s;`
where s is arbitrarily complex code, what statement inside of s will force c to be in a basic block separate from the last one of s ?
(A) `break`
(B) `continue`
(C) `return`
(D) `while`
- For a grammar $G = \langle V_N, V_T, P, S \rangle$, where $V = V_N \cup V_T$. The set of productions P consists of rules of the form $(A \rightarrow \alpha)$, where :
(A) $A \in V$ and $\alpha \in V^*$.
(B) $A \in V_N$ and $\alpha \in V^*$.
(C) $A \in V_T$ and $\alpha \in V_N^*$.
(D) $A \in V^+$ and $\alpha \in V^+$.
- The C++ method dispatch `a->f(x,y)` might be translated into C as :
(A) `(a->class->f)(x,y)`
(B) `(a->class->f)(a,x,y)`
(C) `(a->f)(a,x,y)`
(D) `(**a)(a,x,y)`
- Which of the following items in a state was entered during the closure operation ?
(A) $E \rightarrow \bullet E + T$
(B) $E \rightarrow E \bullet + T$
(C) $E \rightarrow E + \bullet T$
(D) $E \rightarrow E + T \bullet$
- The grammar
 $A \rightarrow x A$
 $A \rightarrow$
(A) is LR(0) but not SLR(1).
(B) is SLR(1) but not LR(0).
(C) is both LR(0) and SLR(1).
(D) is neither LR(0) nor SLR(1).
- According to Sir Thomas Malory's *Le Morte d'Arthur*, who fought "an horrible and a fiendly dragon, spitting fire out of his mouth" ?
(A) Arthur the King
(B) Guinevere the Queen
(C) Lancelot the Knight
(D) Merlin the Wizard

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		$\times 1 =$	$= a$
number of wrong answers		$\times \frac{1}{2} =$	$= b$
number of missing answers		$\times 0 =$	0
column total $c = \max(a - b, 0)$	11		$= c$

- Which (**flex**) pattern will recognize a double-slash comment as in Java or C++ ?
 (A) `/\[/^/n]*`
 (B) `/\[/^\n]*`
 (C) `\[/^/n]*`
 (D) `\[/^\n]*`
- An example of strength reduction is :
 (A) Replacing $(a*1)$ by (a)
 (B) Replacing $(a*32)$ by $(a<<5)$
 (C) Replacing $(3*4)$ by (12)
 (D) Replacing $(a*32)$ by $(a>>5)$
- An example of constant folding is
 (A) Replacing $(a*1)$ by (a)
 (B) Replacing $(a*32)$ by $(a<<5)$
 (C) Replacing $(3*4)$ by (12)
 (D) Replacing $(a*32)$ by $(a>>5)$
- What part of a compiler detects mismatched parentheses ?
 (A) scanner
 (B) parser
 (C) symbol table
 (D) code generator
- Given the grammar fragment presented here, what declarations will go in section 1 of a **bison** grammar that will resolve the ambiguity in the same way as in C and Java ?
 (A) `%left '*'`
 `%left '+'`
 (B) `%left '*' '+'`
 (C) `%left '+'`
 `%left '*'`
 (D) `%left '+' '*'`
- In constructing Follow sets from an LR(0) CFSM, for each state with a (a) access symbol, add to the Follow set of that symbol, any reduction actions in that state, and any (b) symbols on outgoing transitions.
 (A) (a) = nonterminal ; (b) = nonterminal.
 (B) (a) = nonterminal ; (b) = terminal.
 (C) (a) = terminal ; (b) = nonterminal.
 (D) (a) = terminal ; (b) = terminal.
- Type type **YYSTYPE** in code used with **bison** refers to which stack ?
 (A) lexical
 (B) semantic
 (C) state
 (D) symbol
- To resolve the shift/reduce conflict in the following grammar in the same way as C or Java expects, we should insert the declaration (a) into section 1 of a **bison** grammar, because we want to (b).

```

stmt : IF '(' expr ')' stmt ELSE stmt
      | IF '(' expr ')' stmt %prec ELSE
      | whatever
    
```

 (A) (a) = `%left ELSE` ; (b) = reduce.
 (B) (a) = `%left ELSE` ; (b) = shift.
 (C) (a) = `%right ELSE` ; (b) = reduce.
 (D) (a) = `%right ELSE` ; (b) = shift.
- Which of these grammars is ambiguous ?
 (A) $A \rightarrow A + A$
 $A \rightarrow x$
 (B) $A \rightarrow A + x$
 $A \rightarrow x$
 (C) $A \rightarrow x + A$
 $A \rightarrow x$
 (D) $A \rightarrow x + x$
 $A \rightarrow x$
- The set of languages recognizable by an NFA is _____ the set of languages recognizable by a DFA.
 (A) a subset of
 (B) a superset of
 (C) identical to
 (D) none of the above
- What was the first programming language to be specified using Backus-Naur format (BNF) ?
 (A) Algol 60
 (B) COBOL
 (C) FORTRAN
 (D) Pascal