CS 6901 Capstone Exam Systems Winter 2016: Choose any 2 of the 3 problems.

1) Rewrite $F(a, b, c, d) = \sum(1, 3, 5, 7, 11, 12, 13, 14, 15)$ in fully simplified product-of-sums form.

2) Consider the semaphore solution to solve the producer/consumer problem with a buffer of n elements. Write the basic code for producers and consumers. Declare and initialize all semaphores.

3) Consider the following page reference string for a virtual memory system in which physical memory has exactly 3 frames:

5, 1, 2, 1, 5, 8, 1, 6, 8, 5, 6

For each of the following page replacement algorithms, show which references will cause page faults and show the contents of the 3 frames at the time of each fault. Assume that the frames are initially empty. You do not need to show the first 3 faults that are caused by demand paging.

a) Least Recently Used
b) Second Chance

CS 6901 Capstone Exam Data Structures and Algorithms Winter 2016

Choose any 2 of the 3 problems.

1) Consider an application that requires a task to perform the following sequence of operations on a single data structure:

i) find a specific value in the data structure
ii) delete this value from the data structure
iii) insert some other value, at the appropriate place, into the data structure

Consider each of the following data structures as a candidate for the above task, which involves performing all three operations.
For each data structure, state the overall time complexity using big-theta notation in terms of the average case and the worst case as a function of n, where n is the number of elements in the data structure.
Do NOT guess: +2 points for correct answer, -2 points for incorrect answer, 0 points for no answer.

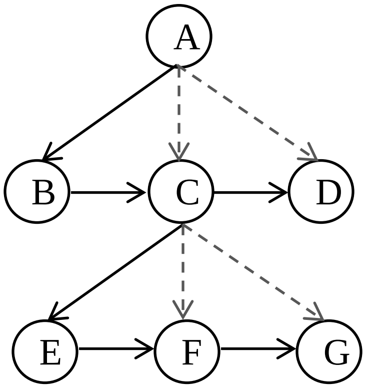
	Big-Theta Notation	
	Average Case	Worst Case
Unsorted Array		
Sorted Array		
Doubly-Linked List		
Binary Search Tree		
Hash Table		

2) Implement a stack of integers using a linked list. Declare the data structure and give code for the following operations:
a) initialize_to_empty
b) push
c) pop (pop should both return a value and remove it from the stack)

3) Consider a general tree containing character values, where a node can have zero or more children. An efficient data structure, with respect to storage, is for each node to have a child pointer (first_child) and a sibling pointer (next_sibling).

For example, the tree below has a root A which has three children B, C, and D. Essentially, the sibling pointers form a linked list of children of one parent.

If the child pointer is NULL, then the node has no children.
If a sibling pointer is NULL, then this is the last child in the linked list.
The root does not have any siblings.



a). Declare the data structure in the language of your choice. For example,

```
typedef struct tree_node *tree_ptr;
……
```

b). Write a recursive function to print out the elements of the tree in a postorder traversal. For example,

```
void postorder(tree_ptr p) {
……
}
```

c). What would be the printout for the tree above?

CS 6901 Capstone Exam Theory Winter 2016

Choose any 2 of the 3 problems.

1. a. State the Pumping Lemma for Context Free languages.

b. Prove that the following language is not context-free:

$L = \{0^m 1^n 0^p : m \geq n \geq p \geq 0\}$

2. a. Define clearly but completely: Language L is decidable.

b. Let $ALL_{DFA} = \{M: M \text{ is a DFA and } L(M) = \Sigma^*\}$ — that is, this is the language of deterministic finite automata that accept all strings. Show that ALL_{DFA} is decidable.

3. If G is an undirected graph, then a vertex cover of G is a subset of the nodes where every edge of G is incident to at least one of those nodes. (An edge and a vertex on that edge are called incident.) The vertex cover problem asks whether a graph contains a vertex cover of a specified size: VERTEX-COVER = {<G, k>: G is an undirected graph that has a k-node vertex cover}.

a. Give a yes-instance and a no-instance of the vertex cover problem.

b. Show that VERTEX-COVER is NP-complete.

(A list of problems that are known to be NP-complete: SAT, 3-CNF-SAT, CLIQUE, HAMPATH, HAMILTONIAN-CIRCUIT, SUBSET-SUM, 3-Dim-Matching, PARTITION, INDEPENDENT-SET.)