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

1) Rewrite

$$F(a, b, c, d) = a'b'c'd + a'bc'd + ab'c'd + ab'cd + abc'd' + abc'd + abcd' + abcd$$

in fully simplified product-of-sums form.

2) Consider two CPU scheduling algorithms for a single CPU: Preemptive Shortest-Job-First (also known as Shortest Remaining Time First) and Round-Robin. Assume that no time is lost during context switching. Given four processes with arrival times and expected CPU time as listed below, draw a Gantt chart to show when each process executes using

a) Preemptive Shortest-Job-First (Shortest Remaining Time First).

b) Round-Robin with a time quantum of 4. For this round-robin trace, calculate the average turnaround time.

Of course, assume that the expected time turns out to be the actual time.

Process	Arrival Time	Expected CPU Time
P1	0	8
P2	2	7
P3	3	4
P4	5	6

3) Consider the following attempted solution to the 2-process mutual exclusion problem.

common variables: flag1, flag2 (both initially false)

Process 1

```
while (true) {
  while (flag2); //empty body
  flag1 = true;
  Critical section;
  flag1 = false;
  Noncritical section;
}
```

Process 2

```
while (true) {
  flag2 = true;
  while (flag1); //empty body
  Critical section;
  flag2 = false;
  Noncritical section;
}
```

a) Does the code guarantee mutual exclusion? If ‘yes’, give a brief explanation of why mutual exclusion must always hold. If ‘no’, give an execution sequence where mutual exclusion is violated.

b) Could deadlock occur? If ‘yes’, give an execution sequence that leads to deadlock. If ‘no’, give a brief explanation of why deadlock is not possible.

c) Is indefinite postponement possible? If ‘yes’, give an execution sequence that results in indefinite postponement. If ‘no’, give a brief explanation of why indefinite postponement is not possible.

CS 6901 Capstone Exam Data Structures and Algorithms Fall 2016:

Choose any 2 of the 3 problems.

1) Consider the implementation of a closed hash table a[0]..a[n-1] to store positive integers, using quadratic probing to resolve collisions. A value of 0 indicates that a hash table location is currently unused. The hash function is $h(x) = x \% n$.

Write a function that is given a new entry x to be inserted. The function returns the index of where it's placed in the array. Return -1 if no empty slot is found. The average runtime of your routine should be according to the usual hashing standards.

2) Write a recursive function that prints out the items of a (possibly empty) singly linked list of integers in reverse order. The function should run in linear time.

For example, given the linked list

83 --> 9 --> 74 --> 122 ,

the output would be 122 74 9 83.

3) Solve the recurrence relation $T(n) = 2T(n/2) + 3n$ where $T(1) = 1$ and $n = 2^k$ for a nonnegative integer *k*. Your answer should be a precise function of n in closed form. An asymptotic answer is not acceptable. Justify your solution.

Theory Exam

Answer **ANY TWO** of the following three questions:

1. Convert the following context-free grammar into Chomsky normal form (CNF):

   $$S \rightarrow S1B \mid C$$
   $$A \rightarrow 0 \mid \varepsilon$$
   $$B \rightarrow AA \mid AC$$
   $$C \rightarrow 0 \mid 11$$

2. In graph theory, an ***independent set*** is a set S of vertices such that for every two vertices in S, there is no edge connecting the two.

   Let INDEPENDENT-SET = {*G*, *k*: *G* is an undirected graph with an independent set of size *k*}.

   Show that INDEPENDENT-SET $\in$ NP.

3. Let PAL_{TM} = {*M* : *M* is a Turing machine that accepts <u>only</u> palindromes}.

   Show that $A_{TM} \leq PAL_{TM}$.