CS 6901   Capstone Exam Systems Fall 2012: Choose any 2 problems.

1) Show the circuit diagram for a 5-bit counter that decrements the stored value on each clock pulse. Use T or JK flip-flops.

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

1, 0, 6, 0, 1, 2, 0, 4, 2, 1

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) Optimal
c) Second Chance

3) Consider the following attempted solution to the 2-process mutual exclusion problem. In addition to the 2 processes, there is also a special process which executes the following code:

```
int p = 1;      // global variable shared by all; initialized once
int lock = 0;   // global variable shared by all; initialized once
while (true) {
  while (lock == 1);  //empty loop body
  if (p == 1) p = 2; else p = 1;
}
```

The code for the 2 processes is as follows:

```
   Process 1
while (true) {
  while (p != 1); //empty body
  lock = 1;
  Critical section;
  lock = 0;
  Noncritical section;
}
```

```
   Process 2
while (true) {
  while (p != 2); //empty body
  lock = 1;
  Critical section;
  lock = 0;
  Noncritical section;
}
```

For each question, answer yes/no with a brief justification.
a) Does the code guarantee mutual exclusion?
b) Is it possible that both processes will busy-wait forever? That is, could deadlock occur?
c) Does the code guarantee fairness? That is, is indefinite postponement impossible?

CS 6901 Capstone Exam Data Structures Fall 2012: Choose any 2 problems.

1. Consider the implementation of a closed hash table of size N to store M items of SomeType, using linear probing to resolve collisions (store in next available free space). The function ***hash(x)*** is given to you; where *hash*(SomeType x) returns a value , $0 \leq$ value $\leq$ (N-1).
a. Write the declaration of the hash table and any code that is necessary to initialize the table.
b. Write the function ***find*** that takes an item y of SomeType and returns its index in the table. If it is not in the table, it returns -1.
c. In storing M items in the table of size N, how many collisions occur in the worst case? How many fewer collisions (worst case) will there be if you change the size of the table to 2N?

2. Write an efficient routine to sort an array of ints a[0] … a[n-1] by the following 2-step process:
Place elements of the array into a dynamically allocated binary search tree.
Write the elements from the tree back into the array in ascending order.

3. Count the precise number of "fundamental operations" executed in the following code. Your answer should be a function of n $(n \geq 2)$ in closed form. Note that "closed form" means that you must resolve all $\sum$'s and $\cdots$'s. An asymptotic answer (such as one that uses big-oh, big-theta, etc.) is not acceptable.

```
for(i = 0; i <= n-2; i++) {
  for(j = 1; j <= n; j++)
    Perform 1 fundamental operation;
  //endfor j
  for (k = i+2; k < n; k++)
    Perform 1 fundamental operation;
  //endfor k
}//endfor i
```

CS 6901 Capstone Exam Theory Fall 2012: Choose any 2 problems.

1. a. Write a Context Free Grammar (CFG) for the language of nonempty data files – described below.

A nonempty *data file* consists of one or more *records*, where each record is one or more *fields*. Each field is either *integer* (one or more digits) or *string* (one or more alphabetic characters enclosed in double quotes).

Every record (including the last one) ends with a period.

Every field (except the last one in a record) ends with a semicolon.

For simplicity, you may assume that the only digits are {0,1,2} and the only alphabetic characters are {a,b,c,d,e}. That is, Σ = {**;** , **.**, ″, 0, 1, 2, a, b, c, d, e}

Example data file with 3 records: 0210**; “a**bc″**;**″a″**.**20111**;** ″bed″**;** ″baba″**;** ″cade″**;** 21**.**″abc″**.**

b. Is your grammar ambiguous? Support your answer.

2. a. Describe carefully the relationships between the languages below using the $\subseteq$ operator. That is, your answer should look like $A \subseteq B \subseteq C$ or $A \subseteq B$, $A \subseteq C$ etc.

TD = set of Turing Decidable (recursive) decision problems
TA = set of Turing Acceptable (recognizable, recursively enumerable) decision problems
NP = set of decision problems that have nondeterministic polynomial Turing Machines
NPC = set of NP-complete decision problems
P = set of decision problems with polynomial solvers

b. For each problem below, give the most restrictive (smallest) class that it belongs to.

L_1 = { <M, w> | M is a Turing machine and w is a string and M *accepts* w}
L_2 = { <M, w> | M is a Turing machine and w is a string and M *does not accept* w}
L_3 = { <M> | M is a nondeterministic finite automaton and $L(M) \neq \emptyset$ (that is, M accepts at least one string)}
L_4 = { <G> | G is a context free grammar and G is *ambiguous* (some string has two parse trees)}
L_5 = { <G> | G is a *connected* graph (no isolated vertices)}
L_6 = { <G, n> | G is a graph with *Hamiltonian circuit* (simple circuit v_1 back to v_1; visits every vertex once)}
L_7 = { N | N is a positive integer and N is *prime* (no divisors except 1 and N)}

3. Choose TWO of the theorems below and give their proofs

(i) If L_1 and L_2 are regular languages, then so is L_1L_2
(ii) If L_1 is a context free language, then so is L_1*
(iii) If L_1 and L_2 are Turing decidable languages, then so is $L_1 \cap L_2$
(iv) If L_1 and L_2 are Turing acceptable languages, then so is $L_1 \cup L_2$