CS 692 Capstone Exam Theory Fall 2018

Choose any 2 of the 3 problems.

1). Let A be the language of all palindromes over {0, 1} containing equal numbers of 0's and 1's. Is A context free? If yes, then give the corresponding context free grammar for A. If not, please prove it. (Palindrome is a string that reads the same backward as forward, for example, 0110.)

2). Give the state diagram for a Turing machine that decides the following language B = {w : w does not contain the substring "ab"} over $\Sigma = \{a, b\}$.

3). a. Completely and briefly define the class "P".

b. Is this true or false? "If $A \leq_p B$ and $B \in P$, then $A \in P$." Please prove your answer.

CS 692   Capstone Exam   Algorithms   Fall 2018: Choose any 2 of the 3 problems.

1) Write a boolean function that is given a binary tree and returns true if and only if the tree has an odd number of nodes. An empty tree is considered to have an even number of nodes.
Notes:
The function should have just one argument, a pointer to the root.
No global variables may be used.
No additional functions may be defined. You may not count the number of nodes.

2) Consider the following insertion sort algorithm.

```
void insertion_sort(element a[], int n)
// Put a[0]..a[n-1] into ascending order by insertion sort.
{
  for (int k = 1; k < n; k++) {
    // At this point, a[0]..a[k-1] are already in order.

    // Insert a[k] where it belongs among a[0]..a[k].
        You need to write code for this insertion as the body
        of the for-k loop.

  }//endfor k
}
```

a) Write the code for the body of the for-k loop to complete the insertion sort routine.
b) Count the precise best case and worst case number of “element comparisons” in your insertion sort routine. Your answers should be functions of n in closed form. Note that “closed form” means that you must resolve all sigmas and …’s. Asymptotic answers (such as ones that use big-oh, big-theta, etc.) are not acceptable.

3) For each function with input argument *n*, determine the asymptotic number of "fundamental operations" that will be executed. Note that `fc` and `fd` are recursive. Choose each answer from among the following. You do not need to explain your choices.

$\theta(1)$ $\theta(\log n)$ $\theta(n)$ $\theta(n \log n)$ $\theta(n^2)$ $\theta(n^2 \log n)$ $\theta(n^3)$ $\theta(2^n)$ $\theta(n!)$

a)
```
void fa(int n) {
  for(i = n; i > 0; i = i/2)
    Perform 1 fundamental operation;
  //endfor i
}
```

b)
```
void fb(int n) {
  for(i = 1; i <= n; i++) {
    for(j = 1; j < n; j++)
      Perform 1 fundamental operation;
    //endfor j
    for(k = 1; k <= i; k++)
      Perform 1 fundamental operation;
    //endfor k
  }//endfor i
}
```

c)
```
void fc(int n) {
  if (n > 1){
    fc(n/2);
    fc(n/2);
    Perform n-1 fundamental operations;
  }//endif
}
```

d)
```
void fd(int n) {
  if (n > 1){
    fd(n/2);
    Perform n-1 fundamental operations;
  }//endif
}
```

CS 692 Capstone Exam Systems Fall 2018: Choose any 2 of the 3 problems.

1) Consider a system with 3 resources (A, B, C) in quantity (7, 6, 7). The Banker's Algorithm is used to allocate resources and it has the following SAFE state:

Process	Allocation			Max			Need		
	A	B	C	A	B	C	A	B	C
P0	1	0	2	2	1	6	1	1	4
P1	0	1	0	2	2	1	2	1	1
P2	3	1	2	6	6	5	3	5	3
P3	0	2	2	0	5	3	0	3	1

Available:	A	B	C
	3	2	1

a) Justify why the current state is safe.
b) If P0 requests an additional unit of resource B, will it be allowed? Justify your answer.

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) Round-Robin with a time quantum of 4.
b) Preemptive Shortest-Job-First (Shortest Remaining Time First).
For part b) only, calculate the average turnaround time.

Process	Arrival Time	Expected CPU Time
P1	0	7
P2	3	9
P3	5	6
P4	9	3

3) There are 3 standard goals to the 2-process mutual exclusion problem:
Goal 1: Mutual exclusion is guaranteed
Goal 2: Deadlock cannot occur.
Goal 3: Indefinite postponement cannot occur.

Attempted Solution: 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;
}
```

For the above solution,
a) Select one goal that is not satisfied and provide an execution sequence that violates the goal.
b) Select one goal that is satisfied and give a brief explanation that justifies why the goal is met for all possible execution sequences.