國立臺南大學104學年度 **資訊工程學系碩士班** 招生考試 計算機概論 試題卷

1. Briefly describe the following terms (15 points)
   (a)Cloud Computing (b) Internet of Things (c)Mobile Computing
   (d)Location-Based Service (e) Multi-Threading

2. Write the following functions in *C* language.
   (a) Write a C program that prints a one-month calendar, and be sure to use **LOOP** to implement your code. A user specifies the number of days in the month and the day of the week on which the month begins: (10 points)

   Enter number of days in month: 31
   Enter starting day of the week (1=Sun., 7= Sat.): 3

   ```
            1   2   3   4   5
    6   7   8   9  10  11  12
   13  14  15  16  17  18  19
   20  21  22  23  24  25  26
   27  28  29  30  31
   ```

   (b) Write a recursive function, ***int Fib(int n)***, to compute and return the Fibonacci number f(*n*) = f(*n*-1)+f(*n*-2), where *n*≥2, f(0)=0, and f(1)=1. (10 points)

3. Write a simple object-oriented program (either in JAVA or C++) containing one super class and its one subclass to show the techniques of *data encapsulation (information hiding)*, *overloading*, and *overwriting*. Write down any constructors as necessary. Note that the length of your code will be part of the grading, and therefore, be sure to keep it as short as possible. In your code, you should provide necessary comments to briefly explain your process. (15 points)

4. Consider a sorted circular doubly-linked list with *n* elements where the head element points to the smallest element in the list. (10 points)
(a) What is the asymptotic time complexity of finding the smallest element in the list?
(b) What is the asymptotic time complexity of finding the largest element in the list?
(c) What is the asymptotic time complexity of finding the median element in the list?
(d) What is the asymptotic time complexity of determining (searching) whether a given element *k* appears in the list?
(e) What is the asymptotic time complexity of deleting a given element *k* in the list (not including the cost of finding it)?

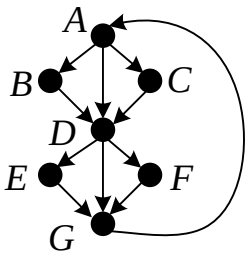
5. Consider the problem for computing a^n, where *a* is an integer number and *n* is a

nonnegative integer.

(a) Design a recursive algorithm for computing 2^n (herein a=2) for any nonnegative integer n that is based on the formula: $2^n=2^{n-1}+2^{n-1}$. (5 points)
Set up a recurrence relation for the number of additions mad by the algorithm and solve it. (3 points)
Please describe its big-O time complexity. (2 points)

(b) For the same problem to computing a^n, please design an algorithm with time complexity $O(\log n)$. Additionally, please show the example for computing 2^{11}, where a=2 and n=11. (10 points)

6. Consider the following directed graph.



(a) List the nodes in the order they would be visited in a depth-first search of the graph starting at *A*. When choosing a node to explore next, break ties in favor of the alphabetically least. (5 points)

(b) List the nodes in the order they would be visited in a breadth-first search of the graph starting at *A*. When choosing a node to explore next, break ties in favor of the alphabetically least. (5 points)

(c) How many directed simple cycles does this graph have? (3 points)

(d) If this graph is an undirected graph, please show its articulation point(s) if any or none. (2 points)

7. For hashing, chaining and probing are two methods used to resolve collisions in hash tables so that the amortized access time is $O(1)$. For each of the following claims (questions), indicate whether it is true of **chaining**, **probing**, **both**, or **neither** method(s). (5 points)

(a) Needs additional memory beyond the primary array for the hash table.

(b) May be either “linear” or “quadratic”.

(c) Requires doubling the table periodically.

(d) Requires computing the hash function multiple times when doing an insertion.

(e) Crashes if the load factor become greater than 1.