國 立 清 華 大 學 命 題 紙

98 學年度 資訊工程學 系（所） _____ 組碩士班入學考試

科目 基礎計算機科學 科目代碼 1901 共 3 頁第 1 頁 *請在【答案卷卡】內作答

1. (5%) Consider x_1, x_2, x_3 and x_4 to be non-negative integers. How many possible solutions are there to the following system of inequalities?

$$x_1 + x_2 + x_3 + x_4 \le 21$$
$$x_1 \ge 1$$
$$x_2 \ge 2$$
$$x_3 \ge 3$$
$$x_4 \ge 4$$

2. (10%)
   (a) (5%) Solve the linear congruence $7x = 13 \pmod{19}$ to find all the integer solutions x.
   (b) (5%) Find the solutions x to the following system of congruences

$$x = 1 \pmod 4$$
$$x = 2 \pmod 5$$
$$x = 3 \pmod 7$$

3. (5%) Show that $1^3 + 2^3 + \cdots + n^3 = (1 + 2 + \cdots + n)^2$.

4. (5%) Consider the following recurrence relation $a_{n+2} - 3a_{n+1} + 2a_n = 4n + 2$, for $n \ge 0$, with $a_0 = 3$, $a_1 = -2$. Find the closed-form solution for a_n.

5. (4%) (a)(2%) List the elements of the set {11, 1010, 100, 1, 101, 111, 110, 1001, 10, 1000} in lexicographic order, given $0 \le 1$.
   (b)(2%) Repeat part (a) assuming $1 \le 0$.

6. (6%) For each of the following sequences, determine if there exists a graph whose degree sequence is the one specified. In each case, either draw a graph, or explain why no graph exists.
   (a) (3%) 4, 4, 4, 3, 2
   (b) (3%) 5, 5, 4, 3, 2, 1

7. (10%) (a)(5%) What is the largest possible number of vertices in a graph with 35 edges, all vertices having degree at least 3?
   (b)(5%) Suppose all vertices in a graph G have odd degree k. Show that the total number of edges in G is a multiple of k.

8. (5 %) Prove that in any graph with more than one vertex, there must exist two vertices of the same degree.

9. (10%) Java is a programming language derived much of its syntax from C and C++. Different from C and C++, Java does not support pointer arithmetic. a) (7%) Please explain the reasons behind such design. (Hint: You might need to use "Stacks" in your answer) b) (3%) Without pointer arithmetic ability, can you still implement linked lists in Java?

10. (6%) Which data structure can be used for converting the infix expression into the postfix form? In what situation, such conversion is necessary?

11. (9%) Please select or fill in the proper answer and briefly explain your answers.

(a) (3%)_ _ _ _ _ _ are used extensively at every level of a modern computer system. For example, they can be used in the basic design of an operating system for interrupt handling and operating system function calls.

1) Queues 2) Stacks 3) Heaps 4) Lists

(b) (3%) The complexity of the following function is _ _ _ _ _ _

```
int failure [max_size];
int pmatch(char *s1, char *ky)
{
  int i=0; j=0;
  int sl = strlen(s1);
  int lk = strlen(ky);
  while (i<sl && j<lk)
  {
      if (s1[i]==ky[j])
      {
          i++; j++
      }
      else if (j==0)
          i++;
      elase
          j=failure[j-1]+1;
  }

  return ((j==lk)? (i-lk) : -1);
}
```

(c) (3%) The Fibonacci numbers are defined as: $f_0=0$; $f_1=1$; and $f_i=f_{i-1}+f_{i-2}$ for i>1. It can be used both iterative and recursive approaches to implement. Usually, the _ _ _ _ _ approach is less efficient.

12. (5%) Prove the edge with second smallest weight must be in a minimum spanning tree.

13. (9%) Sort is one of the most fundamental problems in computer sciences, and many results will benefit from its improvements, so that the understanding of this problem is essential. Quick Sort would be the most popular sorting algorithm since it is very fast in practice. However, we need to implement it carefully to avoid some degenerate cases which will lead bad running time.
    a. (3%) One of degenerate cases is that the input contains a lot of equal keys. How bad will be the running time of Quick Sort on this case? Devise a deterministic trick to conquer this degenerate case, and discuss its overhead and time complexity.
    b. (3%) Another degenerate case is that the input is sorted. Could you improve Quick Sort to expected running time O(n lg n) with random number generators?
    c. (3%) Prove the lower bound of sorting problem based on comparison model. Is Quick Sort optimal with respect to comparison model?

14. (6%) Use hashing method to devise a stable sorting algorithm and explain its correctness and time complexity. Describe the details of your algorithm, e.g., hash function and collision resolution technique you used. Choosing an identity function as the hash function is not allowed.

15. (5%) Apply Kruskal's algorithm on the following figure. Show the picked edges of Kruskal's algorithm in order, and the total sum of weight of minimum spanning tree.