THE UNIVERSITY OF THE SOUTH PACIFIC

CS214: Design & Analysis of Algorithms

SCIMS, FSTE

Final Examination
Semester II, 2019

F2F/Blended Mode

Duration of Exam: 3 hours + 10 minutes

Reading Time: 10 minutes

Writing Time: 3 hours

Total Marks: 40

Instructions:

1. Write your answers in the space provided in this Question Paper.
2. Answer all questions. There are 12 questions and all questions are compulsory.
3. This exam is worth 40% of your overall mark. The minimum mark to pass the final exam is 16/40.
4. Total number of pages including this cover sheet is 14.
5. This is a closed book exam. No printed materials and electronic devices are allowed.

ID:______________________________ **Name:**________________________________

SeatNo: _________________________

This page is intentionally left blank

ID: ________________________

1. Java provides ArrayList and LinkedList data structures for data manipulation. What is your preference and why, when the efficiency of an algorithm is your main concern? Justify your choice clearly. (3 marks)

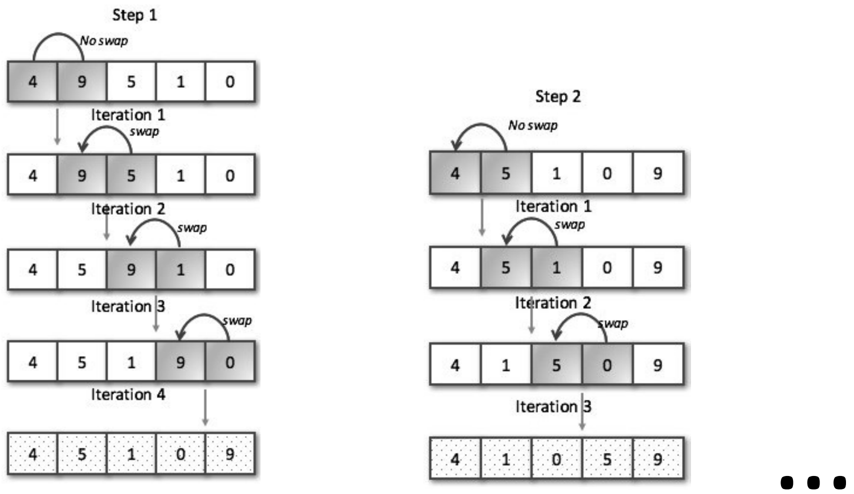
2. A student in CS214 wants to compare the efficiency of two single parameter based functions empirically. How can she approximate their every-case time complexity (if exists)? Assume the parameter is non-negative integer and both functions use same parameter. (3 marks)

ID: ______________________

3. What is the Big O order of the average time complexity for the bubble sort algorithm given below? Show your working. The algorithm and an example illustrating what bubble sort does are shown below. (4 marks)

```
static <T extends Comparable> void bubbleSort (LinkedList<T> S){
    boolean exchanged;
    int n = S.size()+1;
    int counter = 0;
    do {
        n--; //make loop smaller each time
        exchanged = false;
        for ( int i=0; i < n-1; i++ ) {
            if (S.get(i+1).compareTo(S.get(i)) < 0) {
                exchange (S, i, i+1); //exchanges S.get(i) with S.get(i+1)
                exchanged = true; //after exchange must revisit
            }
            counter++;
        }
        System.out.println(S);
    }while (exchanged);

    System.out.println("X(n) = " + counter);
}
```



ID: ________________________

ID: ______________________

4. What is the best case time complexity of the *largest divisible pairs subset* problem given below? Show your working. A working example is also given.

(3 marks)

Problem definition: Given an array of n distinct elements, find length of the largest subset such that every pair in the subset is such that the larger element of the pair is divisible by smaller element.

Algorithm:

```
static int largestSubset(int[] a, int n)
 {
     //the following line has average time complexity of O(log_2 n)
     Arrays.sort(a);

     int[] dp = new int[n];
     dp[n - 1] = 1;

     //Best time complexity: ?
     for (int i = n - 2; i >= 0; i--) {
         int mxm = 0;
         for (int j = i + 1; j < n; j++) {
             if (a[j] % a[i] == 0) {
                 mxm = Math.max(mxm, dp[j]);
             }
         }
         dp[i] = 1 + mxm;
     }

     //the following line has average time complexity of O(n)
     return Arrays.stream(dp).max().getAsInt();
 }
```

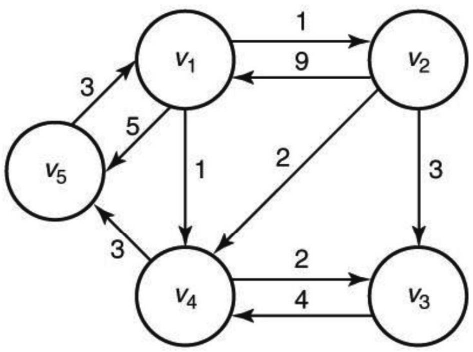
ID: ______________________

5. The floyd's algorithm given below in Figures 1 (a) has generated the path table in Figure 1 (c) from the given graph in Figure 1 (b) to determine the shortest path. What is the shortest path and shortest length from node 3 to node 2? Show your working. (3 marks)

```
void floyd2 (int n,
             const number W[][],
                   number D[][],
                   index  P[][])
{
  index, i, j, k;

  for (i = 1; i <= n; i++)
     for (j = 1; j <= n; j++)
        P[i][j] = 0;
  D = W;
  for (k = 1; k <= n; k++)
     for (i = 1; i <= n; i++)
        for (j = 1; j <= n; j++)
           if (D[i][k] + D[k][j] < D[i][j]){
              P[i][j] = k;
              D[i][j] = D[i][k] + D[k][j];
           }
}
```

(a)



(b)

	1	2	3	4	5
1	0	0	4	0	4
2	5	0	0	0	4
3	5	5	0	0	4
4	5	5	0	0	0
5	0	1	4	1	0

(c)

Figure 1: (a) Floyd algorithm (b) a sample graph (c) path table

6. The Friends Pairing Problem can be written in the following recursive form:

$$f(n) = \begin{cases} f(n-1) + (n-1) * f(n-2), & n > 2 \\ n, & n \leq 2 \end{cases}$$

where n is positive integer.

Problem definition: Given n friends, each one can remain single or can be paired up with some other friend. Each friend can be paired only once. Find out the total number of ways in which friends can remain single or can be paired up.

Would chose dynamic programming or divide & conquer approach to solve this problem? Please explain your reasoning. (2 marks)

7. Suppose preparing a satisfactory USP timetable with deterministic approaches such as dynamic programming, divide and conquer, greedy approach etc. give no better than $O(2^n)$ average case time complexity. How would you solve such problem? Would you be able to produce the optimum (the best possible) timetable? (2 marks)

8. For the given phrase "**tuition at institution**", find its corresponding encoded value using Huffman's algorithm and calculate the ratio of compression achieved. (4 Marks)
   Note: Each character in a phrase occupies 1 byte of memory and ignore the blanks in the given phrase.

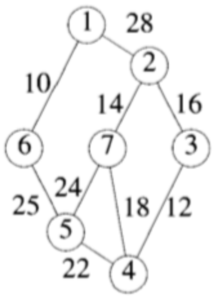
ID: ______________________

9. (a) Write a pseudo code to disperse change from vending machine using greedy approach (2 Marks)

(b) Apply Greedy approach to give a change of **78c** from available coins **54c, 44c, 34c, 14c, 4c, 2c**. Justify whether you have got locally optimal or globally optimal solution. (1 Mark)

ID: ______________________

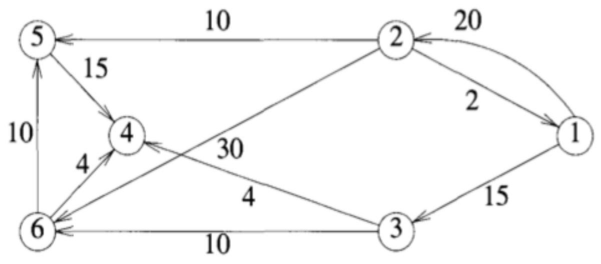
10. (a) Find the minimum cost spanning tree for the given graph using prim's and kruskal's algorithm. Show the output after every stage while using these algorithms (4 Marks)



ID: ________________________

(b) Which is the better of two algorithms for the above graph? Explain your answer. (1 Mark)

11. Use the single source shortest path algorithm to obtain in nondecreasing order the lengths of the shortest paths from vertex 1 to all the remaining vertices in the digraph given below. (2 Marks)



12. Consider the 0-1 Knapsack problem with four different items having different weights and profit values:
    Item 1 (**2** kg, $**3**)
    Item 2 (**3**kg, $**5**)
    Item 3 (**4**kg, $**6**)
    Item 4 (**5**kg, $**10**)
    The knapsack has a capacity of **8**kg

(a) Use the greedy approach to compute the optimal profit in filling the knapsack (1 Mark)

(b) Use the backtracking algorithm for the above 0-1 knapsack problem to maximize the profit and show the state space tree (4 Marks)

ID: ______________________

(c) Which is the better of two approaches above? Explain your answer. (1 Mark)