Final Exam Section2 Fall 2015, Part A
Computer Science Department, SJSU
CS146: Data Structures and Algorithms
Instructor: Katerina Potika

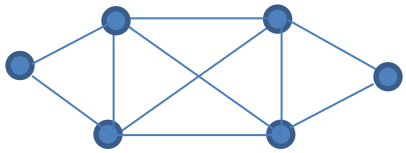
NAME_____SID_____

Section	Points
Section I Multiple Choice 10 questions (2 points each)	_____Out of 20
Section II Short Answer Question 1	_____Out of 21
Section II Short Answer Question 2	_____Out of 18
Section II Short Answer Extra Question	_____Out of 20
Total	_____ Out of 59

Duration 50 min. Closed books. Good Luck!

SECTION I Multiple Choice [2 points each] Choose the best answer for each question.

1. For the following graph what is true:

a. It has a Hamiltonian cycle b. It has an Euler cycle **c. <u>a and b</u>** d. None	

2. What do you know about P=NP
   a. It is true
   b. It is false
   c. <u>It is open</u>
   d. a and c

3. Which problem can be solved optimal by a dynamic programming algorithm
   a. Fractional Knapsack
   b. Single source shortest path
   c. <u>All-pairs shortest path</u>
   d. Clique

4. If you want to prove that problem A is NP-complete which of the following is the correct way to prove it
   a. $Longest\ Path \leq_p A$
   b. $Shortest\ Path \leq_p A$

c. $A \leq_p 3 - SAT$
d. None of the above

5. The distance matrix of a (directed) graph with vertices P,Q, R and S is given by the shortest path from Q to S consists of edges

	P	Q	R	S
P	0	1	∞	2
Q	∞	0	2	5
R	3	∞	0	1
S	1	∞	∞	0

a. **QR and RS**
b. QS
c. QP and PS
d. there is no path

6. state whether the following statement is true or false for a NP-Complete and NP-Hard Problem
   a. if one NP-complete problem , w.l.o.g Clique, has an efficient algorithm then P=NP
   b. To prove NP-hardness of a problem you can only start from SAT or 3SAT.

a**. true, false**
b. true, true
c. false, false
d. false, true

7. Which of the following problems has a polynomial time algorithm
   a. Traveling Salesperson problem
   b. Fractional knapsack
   c. Longest simple path in a graph
   d. Hamilton path

8. Consider the 0-1 knapsack problem. What is the best solution, with 4 items and W=10, such that w=(5, 3, 4, 2) –*weights*- and b=($30, $16, $28, $18) -*values*
   a. 92
   b. 64
   c. 58
   d. none

9. state whether the following statement is true or false for Single Source Shortest Paths

   i. The problem is to determine the shortest paths from a source vertex s to all the remaining vertices of G.
   ii. The way to generate the shortest paths from s to the remaining vertices is to use relaxation.
   iii. If all weights are positive then the solution can be found faster.

   a. **True, true, true**
   b. True, false, true
   c. True, true, false
   d. False, true, false

10. De-duplicate a huge file of objects. Given is a "stream" of objects you can linear scan through a huge file and your goal is to remove duplicates (keep track of unique objects), which is the best data structure to use:
    a. array
    b. red black tree
    c. hash table
    d. linked list

SECTION II – Short Answer– Write a short answer to each question. [20 points each]

Question 1

[3pts/each: total 21pts] Match up each application with an algorithm or data structure that we used to solve it in this course. Use each answer exactly once.

___ range search ___ Document similarity ___ Hamiltonian cycle ___ Mark-sweep garbage collector ___ Web crawler ___ Google maps ___ Semesters for graduating	1. Hashing 2. Red black tree 3. Depth-first search 4. Breadth-first search 5. Dijkstra's algorithm 6. Topological sort 7. Enumerate permutations

Answers:

2 range search 1 Document similarity 7 Hamiltonian cycle 5 Mark-sweep garbage collector 4 Web crawler 5 Google maps 6 Semesters for graduating	1. Hashing 2. Red black tree 3. Depth-first search 4. Breadth-first search 5. Dijkstra's algorithm 6. Topological sort 7. Enumerate permutations

Question 2

For each function on the left, give the best matching order of growth of the running time on the right.

____
public static int f1(int n) {
  int x = 0;
  for (int i = 0; i < n; i++)
  x++;
  return x;
}
 	
____
public static int f2(int n) {
  int x = 0;
	A. O(log n)
	B. O(n)
	C. O(n log n)
	D. O(n^2)
	E. O(2^ n)
	F. O(n!)

```
    for (int i = 0; i < n; i++)
        for (int j = 0; j < i; j++)
            x++;
    return x;
}
```

____

```
public static int f3(int n) {
    if (n == 0)        return 1;
    int x = 0;
    for (int i = 0; i < n; i++)
          x += f3(n-1);
    return x;
}
```

____

```
public static int f4(int n) {
     if (n == 0) return 0;
     return f4(n/2) + f1(n) + f4(n/2);
}
```

____

```
public static int f5(int n) {
     int x = 0;
     for (int i = n; i > 0; i = i/2)
         x += f1(i);
     return x;
}
```

____

```
public static int f6(int n) {
    if (n == 0) return 1;
    return f6(n-1) + f6(n-2);
}
```

```
public static int f7(int n) {
   if (n == 1) return 0;
   return 1 + f7(n/2);
}
```

Answer:

B D F C B E A

Question 2

[24pts] Consider the following 36-character text string: F C F C E C A C B D E D F E A B F B A F F C D C B E D F F F C C D E E F. Find the frequencies of each letter and then compute the Hufmann code for each.

Answer:

A:3
B:4
C:8
D:5
E:6
F:10

Extra question

Suppose that we have a set of activities – lectures have (starting time, finishing time) - to schedule among a large number of lecture halls, where any activity can take place in any lecture hall. We wish to schedule all the activities using as few lecture halls as possible (minimization problem).

a) [10pts] Give an efficient greedy algorithm to determine which activity should use which lecture hall. (This problem is also known as the ***interval-graph coloring problem)***.

We can create an interval graph, i.e. line with times, similar to activity selection problem.

(b) [6pts] Draw the following example, given lectures (1, 4), (2, 5), (6, 7), (4, 8). What is the minimum number of halls that you need?

(c) [4pts] What design technique did you use?

The smallest number of colors required to color every vertex so that no two adjacent vertices have the same color corresponds to finding the fewest lecture halls needed to schedule all of the given activities.)

Answer:

1st

- Use repeated calls to the solution of activity selection algorithm
    - Find a maximum-size set S1 of compatible activities from S for the first lecture hall starting from the beginning,
    - then using it again to find a maximum-size set S2 of compatible activities from S -S1 for the second hall,
    - (and so on until all the activities are assigned), requires Θ(n^2) time in the worst case.
- Is that the optimal (=min #halls)?
- Counterexample: Consider activities with the intervals (1, 4), (2, 5), (6, 7), (4, 8).
    - 1st hall choose the activities (1, 4) and (6, 7) for the first lecture hall, and then each of the activities with intervals
    - Each of (2, 5) and (4, 8) would have to go each into its own hall, for a total of three halls used.
- But optimal solution would put activities (1, 4) and (4, 8) into one

hall and the activities with intervals (2, 5) and (6, 7) into another hall, for only two halls used.

2nd

There is a correct algorithm, however, whose asymptotic time is just the time needed to sort the activities by time

- O(n lg n) time for arbitrary times, or
- possibly as fast as O(n) if the times are small integers.

The general idea is to go through the activities in order of start time, assigning each to any hall that is available at that time. To do this, move through the set of events consisting of activities starting and activities finishing, in order of event time.

Maintain two lists of lecture halls:

1. Halls that are busy at the current event time t (because they have been assigned an activity i that started at s_i <t but won't finish until f_i > t) and

2. halls that are free at time t .

How many halls in a optimal solution?

SCRATCH PAPER