

Final Exam Section1 Spring 2017, Part A  
Computer Science Department, SJSU CS146: Data Structures and Algorithms  
Instructor: Katerina Potika

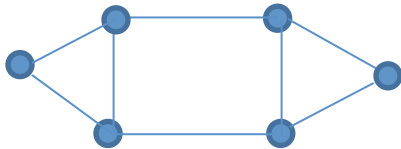
NAME \_\_\_\_\_ SID \_\_\_\_\_

Section	Points
Section I Multiple Choice 10 questions (2 points each)	____ Out of 20
Section II Short Answer Question 1	____ Out of 12
Section II Short Answer Question 2	____ Out of 26
Section II Short Answer Extra Question	____ Out of 8
Total	____ Out of 58

Duration 1h. 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:

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

**2.** What do you know about  $P=NP$

a. It is true   b. It is false   c. It is open   d. a and c

**3.** Which of the following basic algorithms can be used to most efficiently determine the shortest path from a source to all other vertices (if a given graph is connected)?

a. Counting Sort                      c. Postorder traversal  
b. Linear Search                        d. Breadth-First Search

**4.** Consider the following frequencies for each letter A:3, B:4, C:8, D:5, F:10, E:6. What is the Huffman code:

a. A:000 B:001 C:01 D:111 E:110 F:10  
b. A:000 B:001 C:01 D:110 E:111 F:10  
c. A:100 B:101 C:11 D:010 E:011 F:11  
d. None

5. Consider a hash table of size  $m$  and  $n$  keys using chaining for resolving collisions. Under the simple uniform hashing assumption, the *expected* time to insert a new key is at most when you insert  $(\alpha: \text{load factor} - \text{answer } i, ii)$
- In front:
  - In the end:
- a.  $(1,1)$       b.  $(1,n)$       c.  $(1,1+\alpha)$       d.  $(\lg n, \lg n)$

For the next two multiple choice questions consider an instance of the 0-1 Knapsack problem, denote the profit and weight array as  $V$  and  $W$ . Recall that this can be solved by a dynamic programming algorithm by using the following recursive function:

$$A[i-1, j] = \{A[i-1, j], \text{ if } W[i] > j, \text{ otherwise } \max\{A[i-1, j], V[i] + A[i-1, j-W[i]]\}\}$$

Given the result populated table (row/column 0 omitted for simplicity)

	$j = 1$	2	3	4
$i = 1$	1	1	1	1
2	1	1	4	5
3	1	3	4	5

6. What are the profits  $V$  of the items
- $V=[1,4,2,3]$
  - $V=[1,4,3]$
  - $V=[1,3,2]$
  - $V=[3,4,5]$
7. What are the weights of the of the items
- $W=[1,3,2]$
  - $W=[1,4,3]$
  - $W=[1,2,4,5]$
  - None
8. State whether the following statement is true or false for Single Source Shortest Paths
- The problem is to determine the shortest paths between a source node and every other vertex of  $G$ .
  - The way to generate the shortest paths from  $s$  to the remaining vertices is to use edge relaxation.
  - edge weights can only be positive.
- i:True, ii: true, iii: true
  - i:True, ii: false, iii: true
  - i:True, ii: true, iii: false
  - i:False, ii: true, iii: true
9. 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	$\infty$	2
Q	1	0	2	5
R	3	$\infty$	0	3
S	1	$\infty$	$\infty$	0

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

**10.** Say that you decide to upload all of your documents, music, and videos up to the cloud. You are thrilled but also a little worried: how can you be sure that the cloud provider isn't tampering with your files?

- a. Keep copies anyway    c. Print them  
b. Use hash values and store these    d. Don't use the cloud

## SECTION II – Short Answer– Write a short answer to each question.

### Question 1 - Graph Lost in Space [12pts]

Given the following adjacency list, draw the graph, give the visited vertex order for each type of graph search, starting with A, and give the topological sorting or state why one does not exist:

Adj(A) ->B-> D

Adj(B) ->C->E->G

Adj(C) ->A

Adj(D) ->C

Adj(E) ->H

Adj(G) ->F

Adj(H) ->F->G

*Answer*

a. Draw the graph

b. Breadth First Search

*Solution*

c. Depth First Search

*Solution*

d. Does a topological sorting exist?

## Question 2 - Should I stay or should I go [26pts]

Suppose you are a freelancer and that you plan to work at a Mykonos resort for some part of the  $n$ -day summer season next year. Unfortunately, there isn't enough work for you to be paid every day and you need to cover your own expenses (but you want to go). Fortunately, you know in advance that if you are at the resort on the  $i$ th day of the season, you'll make  $p_i$  euros where  $p_i$  could be negative (if expenses are more than earnings) or positive (if expenses are less than earnings). To maximize your earning you should choose carefully which day you arrive and which day you leave; the days you work should be consecutive and you don't need to work all season. For example, if  $n = 8$  and  $p_1 = -9$ ,  $p_2 = 10$ ,  $p_3 = -8$ ,  $p_4 = 10$ ,  $p_5 = 5$ ,  $p_6 = -4$ ,  $p_7 = -2$ ,  $p_8 = 5$  then if you worked from day 2 to day 5, you would earn  $10 - 8 + 10 + 5 = 17$  euros in total. Assume the resort pays your air tickets. Facts: When all numbers are positive, then all days is the answer, but when all numbers are negative; no days are selected and the total is 0. Your result should always be  $\geq 0$ , use arrays.

Describe the algorithms using pseudocode (for loops, recursive calls with parameters etc).

a. [10pts] In the first scenario the day that you arrive is fixed, denote by  $start$ . Can you find your maximum earnings? Describe an efficient algorithm that outputs the maximum earnings or 0. What is the running time of your algorithm?

Solution:

```
int maxEarnings(arr, 1, n, start)
//complete
```

b. [16pts] In the second scenario you can choose the day that you arrive, so your algorithm should return start and maximum earnings. Give a simple  $O(n \lg n)$  divide and conquer algorithm. The idea follows: divide the given array in two halves and return the maximum of the following three max subarray sum in left half, subarray sum in right half and max subarray sum such that the subarray crosses the midpoint.

Fill the pseudocode of the recursive algorithm and the combine step.

**Solution**

```
int maxEarnings (arr, l, h)
```

```
{
```

```
    // Base Case [1pt]
```

```
    // Find dividing point m [1pt]
```

```
    /* Return maximum of following three possible cases
```

```
    a) Maximum subarray sum in left part
```

```
    b) Maximum subarray sum in right part
```

```
    c) Maximum subarray sum such that the subarray crosses the midpoint [8pts]*/
```

```

// Find the maximum possible sum in arr such that arr[m] is part of it [6pts]
int maxCrossingSum(arr, int l, int m, int h)
{
    // Include elements on left of m.

    // Include elements on right of m

    // Return sum of elements on left and right of m

}

```

#### Extra Point Question [8pts]

There are two types of professional chess players: “purple” and “green”. Between any pair of professional players, there may or may not be a rivalry. Suppose we have  $n$  professional player and we have a list of  $r$  pairs of players for which there are rivalries. What algorithm will determine whether it is possible to designate some of the players as purple and the remainder as green such that each rivalry is between a purple and a green? What is the running time?

SCRATCH PAPER