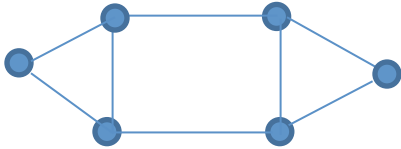


Final Exam Section1 Spring 2017, Part A Key
CS146: Data Structures and Algorithms
Instructor: Katerina Potika

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:

<p>a. <u>It has a Hamiltonian cycle</u></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
 - b. Postorder traversal
 - c. Depth-First 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. The array 1 4 2 5 8 6 10 is organized into a min heap (priority queue). Which array represents the heap after two deleteMin operations have been performed?
- a. 10 8 6 5 4
 - b. 4 6 5 10 8
 - c. 4 5 6 10 8

d. 4 10 6 5 8

6. 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 in the (α : load factor)
- In front:
 - In the end:
- (1,1)
 - (1,n)
 - (1,1+ α)**
 - (lgn,lgn)

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

7. What are the profits V of the items
- $V=[1,4,2,3]$
 - $V=[1,4,3]$**
 - $V=[1,3,2]$
 - $V=[3,4,5]$
8. What are the weights of the of the items
- $W=[1,3,2]$**
 - $W=[1,4,3]$
 - $W=[1,2,4,5]$
 - None
9. You are driving alone from SJ to NY in a Porsche 911 Carrera on a fixed 2942 mile route. Assume that you can go 250 miles on one tank of gas and that you have a map showing every gas station on the route. What technique for designing algorithms would you choose to determine which gas stations to stop at and how much gas to get at each station?
- Divide and conquer
 - Greedy**
 - Dynamic programming
 - None of the above

10. State whether the following statement is true or false for Single Source Shortest Paths
- I. The problem is to determine the shortest paths between a source node and every other vertex of G.
 - II. The way to generate the shortest paths from s to the remaining vertices is to use edge relaxation.
 - III. edge weights can only be positive.
- a. i:True, ii: true, iii: true
 - b. i:True, ii: false, iii: true
 - c. **i:True, ii: true, iii: false**
 - d. i:False, ii: true, iii: true
11. 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	1	0	2	5
R	3	∞	0	3
S	1	∞	∞	0

- a. QR and RS
 - b. QS
 - c. **QP and PS**
 - d. there is no path
12. 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 cloud provider isn't tampering with your files?
- a. Keep copies anyway
 - b. Print them
 - c. **Use hash values and store these**
 - d. Don't use the cloud

Extra Question (5pts)

There are two types of professional wrestlers: “babyfaces” (“good guys”) and “heels” (“bad guys”). Between any pair of professional wrestlers, there may or may not be a rivalry. Suppose we have n professional wrestlers and we have a list of r pairs of wrestlers for which there are rivalries. Give an $O(n+r)$ -time algorithm that determines whether it is possible to designate some of the wrestlers as babyfaces and the remainder as heels such that each rivalry is between a babyface and a heel. If it is possible to perform such a designation, your algorithm should produce it.

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

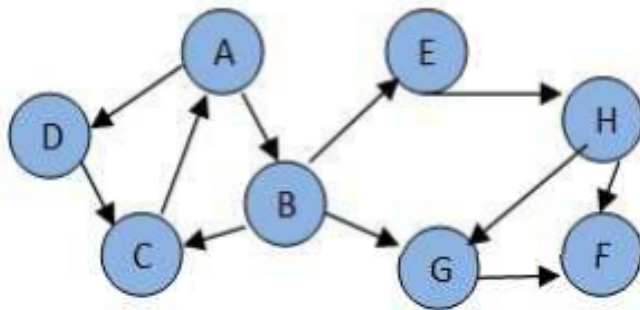
Question 1 [12pts]

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

Adj(A) \rightarrow B \rightarrow D
 Adj(B) \rightarrow C \rightarrow E \rightarrow G
 Adj(C) \rightarrow A
 Adj(D) \rightarrow C
 Adj(E) \rightarrow H
 Adj(G) \rightarrow F
 Adj(H) \rightarrow F \rightarrow G

Answer

a. Draw the graph



b. Breadth First Search

Solution: ABDCEGHF

c. Depth First Search

Solution ABCEHFGD

d. not a DAG \rightarrow no topological sorting

Question 2

(Should I stay or should I go). 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 airtickets.

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 0 or a positive number. Hint: use arrays.

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

[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:

the for loop finds the maximum possible sum with first element in *start* position. The time complexity is $O(n)$.

```
int maxSubArraySumFixed(arr[], n, start)
```

```
    sum=0; //keeps the maximum sum starting at i
    sum_temp=0; //temporary maximum sum
    for (j=i; j<=n; j++)
        sum_temp = sum_temp + arr[j]
        if (sum_temp > sum)
            sum = sum_temp
    if sum<0
        return 0
    else return sum
```

[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.

Algorithm D&C (idea)

Use a Divide and Conquer approach. The idea follows

- 1) Divide the given array in two halves
- 2) Return the maximum of the following three

....a) Maximum subarray sum in left half (Make a recursive call)
b) Maximum subarray sum in right half (Make a recursive call)
c) Maximum subarray sum such that the subarray crosses the midpoint
 For 2c) we can easily find the crossing sum as follows: simply combine the left part and the right part. In our example of array B the maximum on the left 4 (=6-2) and the maximum on the right side is 3 (= -3+1+5).
 Write the pseudocode of the recursive algorithm and the combine step.

Solution:

find the maximum sum starting from mid point and ending at some point on left of mid, then find the maximum sum starting from mid + 1 and ending with sum point on right of mid + 1.

Solution

we can find the maximum subarray sum in $O(n \log n)$ time.

```
int maxSubArraySum(arr[], l, h)
{
    // Base Case: Only one element
    if (l == h)
        return arr[l]

    // Find middle point
    int m = (l + h) / 2

    /* Return maximum of following three possible cases
    a) Maximum subarray sum in left half
    b) Maximum subarray sum in right half
    c) Maximum subarray sum such that the subarray crosses the midpoint */
    return max(maxSubArraySum(arr, l, m),
               maxSubArraySum(arr, m+1, h),
               maxCrossingSum(arr, l, m, h))
}

// Find the maximum possible sum in arr[] such that arr[m] is part of it
int maxCrossingSum(arr[], int l, int m, int h)
{
    // Include elements on left of mid.
    int sum = 0;
    int left_sum = 0;
    for (int i = m; i >= l; i--)
    {
        sum = sum + arr[i]
        if (sum > left_sum)
            left_sum = sum
    }
}
```

```
// Include elements on right of mid
sum = 0
int right_sum = 0
for (int i = m+1; i <= h; i++)
{
    sum = sum + arr[i]
    if (sum > right_sum)
        right_sum = sum
}

// Return sum of elements on left and right of mid
return left_sum + right_sum
}
```

running time: $T(n) = 2T(n/2) + \Theta(n)$