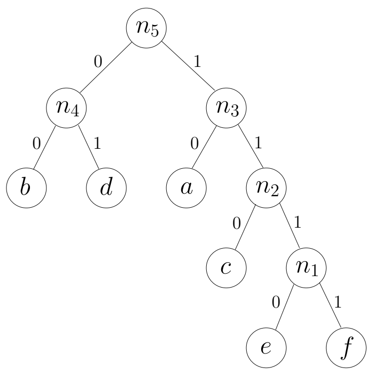
COMP 3711 Design and Analysis of Algorithms
Fall 2015 Final Exam Solutions

1.

case	1	2	3	4	5	6	7	8
Faster	B	A	U	U	U	A	B	A

2. (1) a, (2) b, (3) d, (4) c, (5) d

3.



Huffman code:

$a = 10$

$b = 00$

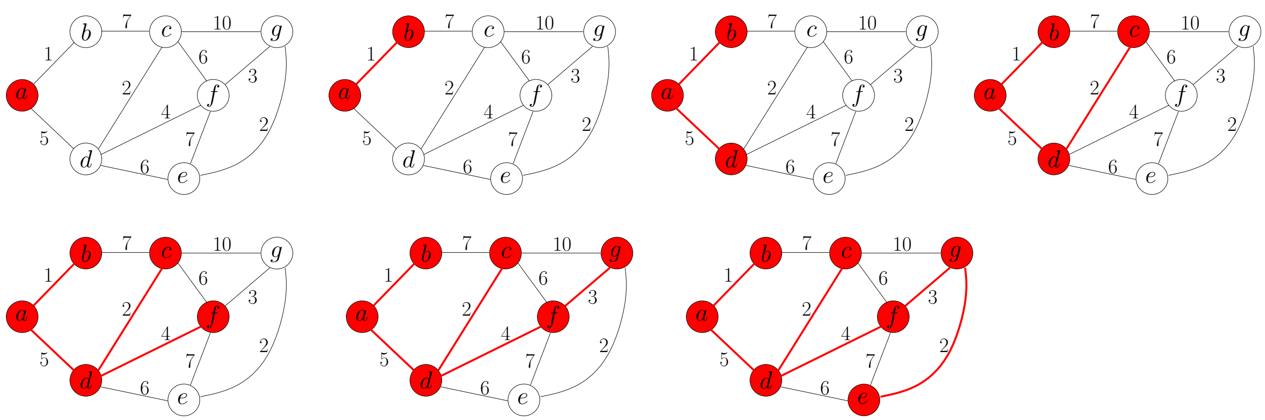
$c = 110$

$d = 01$

$e = 1110$

$f = 1111$

4.



5. **Structure:** For $1 \le i \le n$, define $c[i]$ be the minimum subsequence sum of $A[1..i]$ and $A[i]$ must be included in the subsequence. $\min\{c[n-2], c[n-1], c[n]\}$ is the optimal value of the original problem.

Recurrence:

Base case: $c[1], c[2]$ and $c[3]$ are solved directly.

Recursive case (for $i > 3$): $c[i] = A[i] + \min\{c[i-3], c[i-2], c[i-1]\}$

Bottom-up computation: Compute $c[i]$ in increasing order of i.

Construction of optimal solution: For $1 \le i \le n$, define $d[i] = j$ such that $c[i]$ is obtained from $A[i] + c[j]$ and $d[i] = 0$ if $A[i]$ is the leftmost element in the subsequence.

Output the subsequence by the following procedure:
OutputSubsequence(A, d, i)
1. while $i > 0$ {
2.   output $A[i]$
3. $i = d[i]$
4. }
Initial call: OutputSubsequence(A, d, n'), where $c[n'] = \min\{c[n-2], c[n-1], c[n]\}$.
Running time analyze: There are $O(n)$ subproblems and each subproblem can be solved in $O(1)$ time, so the total running time is $O(n)$.

6. **Structure:** For $1 \le i \le n$, $1 \le j \le n$, define $c[i, j]$ be the DTW of $x[1..i]$ and $y[1..j]$. $c[n, n]$ is the optimal value of the original problem.
Recurrence:
Base case: $c[1, j] = \sum_{k=1}^{j} |x[1] - y[k]|$, $c[i, 1] = \sum_{i=1}^{k} |x[k] - y[1]|$ for $1 \le i, j \le n$
Recursive case: $c[i, j] = |x[i] - y[j]| + \min\{c[i-1, j], c[i-1, j-1], c[i, j-1]\}$.
Bottom-up computation: Compute $c[i, j]$ in increasing order of i and j.
Running time analysis: There are $O(n^2)$ subproblems and each subproblem can be solved in $O(1)$ time, so the total running time is $O(n^2)$.

7. Run the topological sorting algorithm on the graph with the following change: In each iteration, if there is one vertex with in-degree 0, output that vertex; if there is zero or more than one vertex with in-degree 0, output "no Hamiltonian path". Obviously, it runs in $O(V + E)$ time.

Alternative solution: Find the longest path and check if it visits all vertices.

8. Create a source s and add the directed edge (s, u) and (s, w). Each undirected edge in G becomes two anti-parallel edges. Set the target to be $t = v$, and each vertex (except s and v) has capacity 1. Find the max flow of this flow network with source s and target $t = v$ by the technique in WA4. If the max flow is 2, there exists a simple path from u to w that passes through v. The Ford-Fulkerson algorithm to find the max flow of this flow network, the algorithm will stop in at most 2 iterations. So the total running time is $O(V + E)$.

9. Consider the points in decreasing x-coordinates. Let $X_i = 1$ if the i-th point is on the skyline, and $X_i = 0$ otherwise. Observe that $X_i = 1$ iff the i-th point is the highest in the first i points (same as the hiring problem), so $\Pr[X_i = 1] = 1/i$. Then the expected number of skyline points is $\sum_{i=1}^{n} \frac{1}{i} = \Theta(\log n)$.