University of Nevada, Las Vegas Computer Science 477/677 Fall 2023

Answers to Examination October 25, 2023

Name:_____

The entire examination is 240 points.

1. Fill in the blanksj

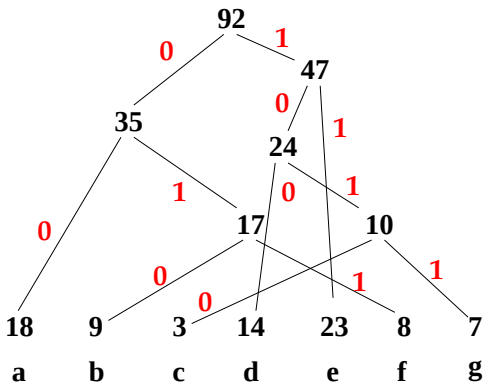
   (a) [5 points] The asymptotic time complexity of the Floyd Warshall algorithm on a weighted directed graph with n vertices and m arcs is $\Theta(n^3)$. (Use Θ.)

   (b) [5 points] The asymptotic time complexity of Dijkstra's algorithm on a weighted directed graph with n vertices and m arcs is $O(m \log n)$. $O(m \log m)$ is also correct.

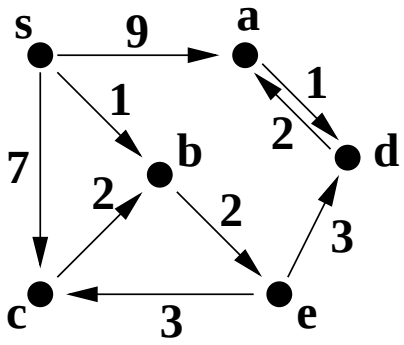
   (c) [10 points] The asymptotic time complexity of Treesort of n items is $O(n^2)$ in the worst case, but if you use a balancing scheme for the tree, you can expect that the asymptotic time complexity is $O(\log n)$.

2. [10 points] Find an optimal prefix-free binary code for the following weighted alphabet.

a	18	00
b	9	010
c	3	1010
d	14	100
e	23	11
f	8	011
g	7	1011



3. [20 points] Use Dijkstra's algorithm to solve the single source minpath problem for the weighted digraph shown below.



s	a	b	c	d	e
0	~~9~~ 8	1	~~7~~ 6	6	3
*	~~s~~ d	s	~~s~~ e	e	b

4. [20 points] Write pseudocode for the Floyd Warshall algorithm on a weighted directed graph. Assume the vertices are the integers from 0 to $n-1$ and W$[i,j]$ is the weight of the arc from i to j. If there is no such arc, W$[i,j] = \infty$. Your code should calculate all V$[i,j]$, the smallest weight of any path from i to j, and back$[i,j]$, the back pointer for that path.

```
for all i from 0 to n-1 and all j from 0 to n-1
  V[i,j] = W[i,j]
  back[i,j] = i
for all i from 0 to n-1
  V[i,i] = 0
for all j from 0 to n-1
  for all i from 0 to n-1
    for all k from 0 to n-1
      if(V[i,j]+V[j,k] < V[i,k])
        V[i,k] = V[i,j]+V[j,k]
        back[i,k] = back[j,k]
```

5. [20 points] Write pseudocode for the Bellman Ford algorithm on a weighted directed graph. I won't specify the notation as I did for Floyd Warshall; I will leave that task to you. Be sure to incorporate the shortcut.

   The vertices are the integers $0 \ldots n-1$, and the source vertex is 0,

   There are m arcs, $a_1, \ldots a_m$. For $1 \leq k \leq m$, e_k is the ordered pair (s_k, t_k), and has weight w_k. We let $V[i]$ be the least weight of a path from 0 to i.

   For all i $V[i] = \infty$
   finished = false
   while(not finished)
     finished = true
     for all k from 1 to m
       if($V[s_k] + w_k < V[t_k]$)
         $V[t_k] = V[s_k] + w_k$
         finished = false

6. [10 points] Solve the recurrence $F(n) = 2F(n/4) + \sqrt{n}$

   $A = 2$, $B = 4$, $C = 1/2 = log_B A$

   $F(n) = \Theta(\sqrt{n} \log n)$

7. [10 points] Solve the recurrence $F(n) = F(n - \sqrt{n}) + n^2$

   $$\frac{F(n) - F(n-\sqrt{n})}{\sqrt{n}} = \frac{n^2}{\sqrt{n}}$$

   $F'(n) = \Theta(n^{3/2})$

   $F(n) = \Theta(n^{5/2})$

8. [10 points] Solve the recurrence $F(n) = F(\log n) + 1$

$F(n) = \Theta(\log^* n)$

9. For each of the following recurrences, substitute $m = \log n$.

   (a) [10 points] $F(n) = F(\sqrt{n}) + 1$

   Let $G(m) = F(n)$, $G(m/2) = F(\sqrt{n})$ $G(m) = G(m/2) + 1$
   $F(n) = G(m) = \Theta(\log m) = \Theta(\log \log n)$

   (b) [10 points] $F(n) = 4F(\sqrt{n}) + 1$

   $G(m) = 4G(m/2) + 1$
   $F(n) = G(m) = \Theta(m^2) = \Theta(\log^2 n$

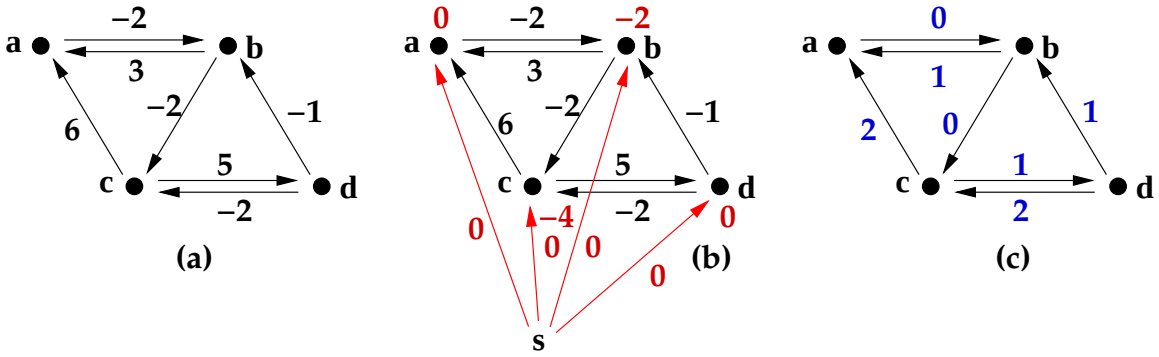
   (c) [10 points] $F(n) = 4F(\sqrt{n}) + \log^2 n$

   $G(m) = 4G(m/2) + m^2$
   $F(n) = G(m) = \Theta(m^2 \log m) = \Theta(\log^2 n \log \log n)$

10. [20 points] You are given a sequence of positive numbers, say $x_1 \ldots x_n$. The problem is to choose a maximum sum legal subsequence, where a subsequence is legal if it does not include two consecutive terms of the original sequence. Design a dynamic programming algorithm which solves this problem in $O(n)$ time.

$M[0] = 0$
$M[1] = x_1$
$M[i] = \max\{M[i-1], M[i-2] + x_i\}$ for $i \geq 2$

11. [20 points] Consider the weighted directed graph shown below in (a). Since there are negative weight edges, the weights of the edges are adjusted, using the method given by Johnson's algorithm. Show the adjusted weights in (c). You can use (a) and (b) for your work.



12. Express each value in the $a + b\,\mathbf{i}$ form. Simplify each expression as much as possible.

    Don't forget that $\mathbf{i}^2 = -1$.

    (a) [10 points]
    $$\frac{3+\mathbf{i}}{2-\mathbf{i}} = \frac{(3+\mathbf{i})(2+\mathbf{i})}{(2-\mathbf{i})(2+\mathbf{i})} = \frac{6+3\mathbf{i}+2\mathbf{i}+\mathbf{i}^2}{4-\mathbf{i}^2} = \frac{5+5\mathbf{i}}{5} = 1+\mathbf{i}$$

    (b) [10 points] $e^{\pi\,\mathbf{i}}$

    $\cos\pi = -1$ and $\sin\pi = 0$

    $e^{\pi\,\mathbf{i}} = \cos\pi + \mathbf{i}\sin\pi = -1$

    (c) [10 points] $(1+\mathbf{i})^8$ (Hint: the answer is a positive real number)

    $(1+\mathbf{i})^2 = \mathbf{2i}$

    $(1+\mathbf{i})^4 = \mathbf{-4}$

    $(1+\mathbf{i})^8 = \mathbf{16}$

13. [20 points] Here is C++ code for quicksort on a given array $A[N]$ of type int, as given in the handout sorting.pdf. I have tested the code.

```
void swap(int&x,int&y)
 {
  int temp = x;
  x = y;
  y = temp;
 }

void quicksort(int first, int last)    // input condition: first <= last
  // sorts the subarray A[first .. last]
 {
  if(first < last) // otherwise there is only one entry
   {
    int mid = (first+last)/2;
    swap(A[first],A[mid]);
    int pivot = A[first];
    int lo = first;
    int hi = last;
     // loop invariant holds
    while(lo < hi) // the partition loop
     {
       // loop invariant holds
      while(A[lo+1] < pivot)lo++;
      while(A[hi] > pivot)hi--;
      if(lo+1 < hi)
       {
        swap(A[lo+1],A[hi]);
```

```
        lo++;
        hi--;
       }
      else if(lo+1 == hi) hi--;
     }
     // loop invariant holds
    swap(A[first],A[lo]);
    // now A[lo] = pivot
    if(first < lo) quicksort(first,lo-1);
    if(lo+1 < last) quicksort(lo+1,last);
   }
 }

int main()
 {
  quicksort(0,N-1);
  cout << endl;
  return 1;
 }
```

What is the loop invariant of the partition loop?

$lo <= hi$ and $A[i] \leq pivot$ for all $first \leq i \leq lo$ and $A[i] \geq pivot$ for all $hi < i \leq last$