University of Nevada, Las Vegas Computer Science 477/677 Spring 2026

Answers to Assignment 6: Due Saturday May 2, 2026

Name:_____

1. Evaluate each of these expressions.

   (i) $\log_2 4 = 2$

   (ii) $\log_5 \left(4\left(\frac{2}{5}\right)^2 + \left(\frac{3}{5}\right)^2 \right) = 0$

   (iii) $3^{log_9(49)} = 7$ (Hint: it's an integer.)

2. (i) What is the asymptotic complexity of $george(n)$, computed by the recursive code below?
   $\Theta(n \log n)$

   (ii) What is the asymptotic time complexity of the computation of $george(n)$ using the code below?
   $\Theta(n)$

   (iii) Asymptotically, how many memos are needed if $george(n)$ is computed using memoization?
   $\Theta(\log^2 n)$

```
int george(int n)
 {
  if(n < 3) return 1;
  return george(n/2) + george(n/3) + george(n/6) + n;
 }
```

3. Fill in the blanks.

   (i) We say that a graph is **connected** if there is a path between any two vertices.

   (ii) A **clique** is a graph where there is an edge connecting any two vertices.

   (iii) A binary tree of height h has at least $h + 1$ nodes, and has no more than $2^{h+1} - 1$ nodes. (Exact answers, which are algebraic expressions, please.)
   (The height of a binary tree is the length of the longest path from the root to a leaf.)

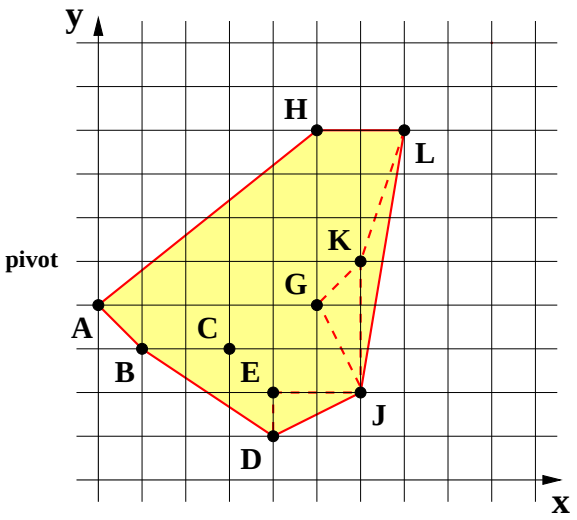
   (iv) A graph G is **planar** if it can be embedded in a plane with no crossings. In that case, if G has n vertices, where $n \geq 3$, then G has no more than $3n - 6$ edges. (Exact answer, which is an algebraic expression.)

4. Use Graham Scan to find the convex hull of the set of points in the Euclidean plane listed below.

$A = (0, 4)$ $G = (5, 4)$
$B = (1, 3)$ $H = (5, 8)$
$C = (3, 3)$ $I = (5, 8)$
$D = (4, 1)$ $J = (6, 2)$
$E = (4, 2)$ $K = (6, 5)$
$F = (4, 1)$ $L = (7, 8)$
F = D and H = I, so I eliminated F and I.

We take A to be the pivot. Here is sorted order of the points in order of the slope of the line segment from A to that point. The point C is deleted, because AC and AJ have the same slope, and C is closer to A than J. We are left with the points B, D, E, J, G, K, L, and H.

We first draw the line AB, then BD, then DE, all left turns. We draw the line EJ, a right turn, so we delete the last two lines (shown as dashed) and draw the line DJ, then JG, then GK. The last step is a right turn, so we draw the line JK and delete JG and GK. We now draw KL, another right turn, so we draw JL and delete KL. Finally, we draw LH and HA, both left turns, and we are done. The entire convex hull is the polygon together with its interior.



5. The array $A[10][5][8]$ is stored in row major order in RAM. $A[0][0][0]$ is stored in RAM[1024] and each entry requires two addresses in RAM. What is the RAM address of $A[8][3][2]$?

   That entry has $8*5*8+1*3*8+1*1*2 = 346$ predecessors, and its RAM address is $1024+346*2 = 692$.

6. $R[][]$ is a virtual array of 8 rows, where the first row has length 1 and each subsequent row is twice the length of the previous row. That is, $R[i, j]$ is an entry if $0 \leq i < 8$ and $0 \leq j < 2^i$. How many entries does R have? $\sum_{i=0}^{7} 2^i = 2^8 - 1 = 255$. How many predecessors does $R[7][45]$ have?

   The predecessors are the $2^7 - 1 = 127$ entries in the first 7 rows plus the first 45 in row 7, a total of 172 predecessors.

7. Find a longest monotone increasing subsequence of the sequence:

   $YDRCTNWQZFMGPHSK$.

   Here are the columns we build during the computation.

```
1  2  3  4  5  6

Y  R  T  W  Z
D  N  Q  P  S
C  F  M  H  K
      G
```

   The list of last items in each column, namely C F G H K, is a longest monotone increasing subsequence.

8. Let $G = (V, E)$ be a graph, where $|V| = n$ and $|E| = m$. Describe an algorithm which computes the components of G quickly. Hint: there is more than one good answer.

   If G is given as an array of neighbors, the components can quickly be identified with either DFS or BFS. If G is given simply as a list of edges, I believe it would be faster to use union/find.

9. Find the asymptotic time complexity of each code fragment, in terms of n. You have plenty of time; get them right!

   (i)
```
for(int i = 1; i < n; i++)
  for(int j = 1; j < i; j = 2*j)
```
   $\Theta(n \log n)$

   (ii)
```
for(int i = 1; i < n; i++)
  for(int j = i; j < n; j = 2*j)
```
   $\Theta(n)$

   (iii)
```
for(int i = 1; i < n; i++)
  for(int j = n; j > i; j = j/2)
```
   $\Theta(n)$

   (iv)
```
for(int i = 1; i < n; i++)
  for(int j = i; j > 1; j = j/2)
```
   $\Theta(n \log n)$

   (v)
```
for(int i = 2; i < n; i = i*i)
```
   $\Theta(\log \log n)$

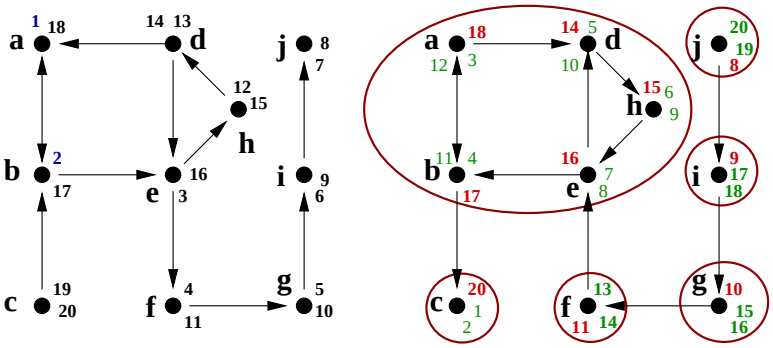
   (vi)
```
for(int i = 1; sqrt(i) < n; i++)
```
   This is tricky. If $\sqrt{i} < n$ then $i < n^2$. The time complexity is $\Theta(n^2)$.

10. The following table lists the out-neighbors of each vertex of a directed graph G. Write the table of in-neighbor lists of G.

a	b
b	a,e
c	b
d	a,e
e	h,f
f	g
g	i
h	d
i	j
j	

a	b
b	a,c
c	
d	h
e	b,d
f	e
g	f
h	b,e
i	g
j	i

11. Compute the strong components of the graph given in problem 10.



12. Huffman's algorithm creates a code which satisfies the *prefix condition.* What is the purpose of that condition?

    While reading the stream of bits, the prefix condition allows the reader to determine the end of a code string.

13. (i) _____ What is the name of the algorithm given by the following code?

```
for(int i = 0; i < n; i++)
  for(int j = i+1; j < n; j++);
    if(A[j] < A[i]) swap(A[i],A[j]);
```

    (ii) The code has two loops. Give the loop invariant of each loop.

    The invariant of the outer loop is that the subarray A[0] ... A[i−1] is sorted. The invariant of the inner loop is that A[i] ≤ A[k] for all i<k<j.

    You can replace each **for** loop with a **while** loop, if that helps:

```
int i = 0;
while(i < n)
 {
  j = i+1;
  while(j < n)
```

```
   {
    if(A[j] < A[i]) swap(A[i],A[j]);
    j = j+1;
   }
  i = i+1;
 }
```

14. A C++ program uses a binary search tree (bst) of characters. Here is the definition of the binary search tree. Note that two types are defined: a struct, "bstnode," and "bst," a pointer to "bstnode."

```
struct bstnode;
typedef bstnode*bst;
struct bstnode
 {
  char item;
  bst lft;
  bst rgt;
 };
```

Complete the C++ code below, for the recursive function which computes the height of a binary search tree.

```
int hite(bst t) // height
  // empty tree has height -1
  // singleton tree has height 0
 {
  if(t)
   return 1+max2(hite(t->left),hite(t->right));
  else return -1;
 }
```

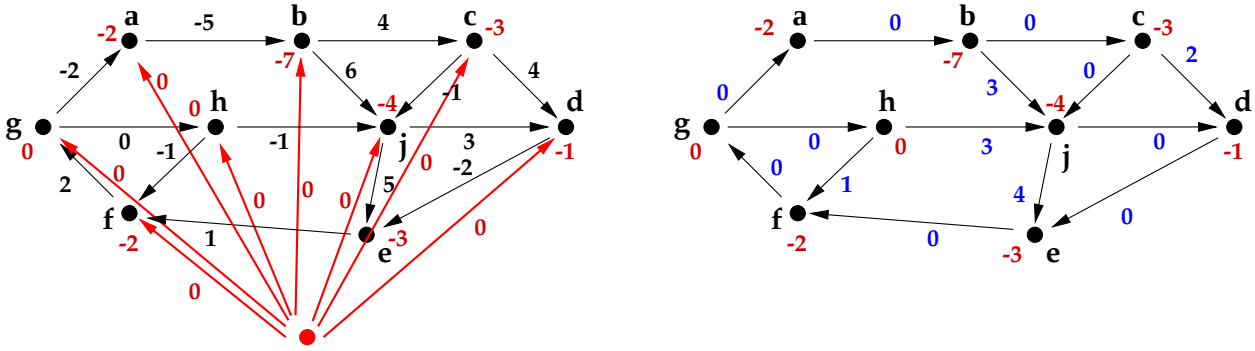
Complete the C++ code below for the recursive function which traverses a binary search tree, printing the characters in alphabetic order, also known as "inorder."

```
void inorder(bst t)
 {
    if(t)
     {
      inorder(t->left);
      cout << t->item;
      inorder(t->right);
     }
 }
```

Complete the C++ code below, for the function which inserts a new character into a bst.

```
void insert(bst&t,char newitem)
 {
  // duplicates not inserted
    if(t==NULL)
     {
      t = new bstnode();
      t->item = newitem;
     }
    else if(newitem < t->item) insert(t->lft,newitem);
    else if(newitem > t->item) insert(t->rgt,newitem);
 }
```

15. Work the first part of Johnson's algorithm on the weighted directed graph shown in (a) below. In (b) show the adjusted weight of each arc. Do not finish Johnson's algorithm.



16. All the sorting algorithm we've studied this semester use the comparison model of computation, but *radix sort* does not. Describe radix sort, where the list being sorted consists of binary strings of length n.

    The sorting algorithm consists of n phases. For k from n to 1, the phase sorts the list by the k^{th} bit. That sorting is *stable*, meaning that the relative order of two strings which have the same k^{th} bit is maintained.

17. Give the *information theory* proof that the time complexity of any sequential algorithm for sorting a list of n items which uses the comparison model of computation, is $\Omega(n \log n)$ in the worst case.

    Let $\mathcal{A}$ be a sorting algorithm which uses the comparison model of computation. Without loss of generality, the items sorted are the integers from 1 to n. An input to $\mathcal{A}$ must consist of those integers in some order, which is a permutation σ of $\{1 \ldots n\}$. Let π_σ be the path the computation, given input σ, through the flow chart of $\mathcal{A}$, That computation must execute a permutation on the list, and that permutation is the inverse of σ. Thus $\pi_{\sigma 1} \neq \pi_{\sigma 2}$ if $\sigma 1$ and $\sigma 2$ are different paths through the flow chart.

    Let t be the worst case number of comparisons in one computation of $\mathcal{A}$. Each path contains at most t comparisons, hence there are at most 2^t paths through the flow chart. Each choice of permuation requires a different computation, hence there must be at least $n!$ such paths. Hence $2^t \geq n!$ Taking the

base 2 logarithm of both sides, we have $t \geq \log_2 n!$. Since $n! = \prod_{i=1}^{n} i$, $\log_2 n! = \sum_{i=1}^{n} \log_2 i$, which is asymptotically the same as $\int_1^n \ln x dx = \Theta(n \log n)$. Hence $t = \Omega(n \log n)$.

18. Write pseudocode for an algorithm which finds the greatest sum of a contiguous subsequence of any sequence $\sigma = x_1, x_2, \ldots x_n$ of numbers. The σ may contain negative as well as positive numbers. For example,if σ is 3,-4,2,6,-3,2,-8,4,7,-2,6,-1,8,-5, the maximum weight contiguous subsequence of σ is 4,7,-2,6,-1,8, which has weight 22. Your algorithm should take $O(n)$ time.

    We define two arrays, A and B. Let $A[i]$ be the maximum weight of any contiguous subsequence of $\sigma = x_1, \ldots x_i$, defaulting to 0 if $x_j < 0$ for all $j <= i$, and let $B[i]$ be the maximum weight of any contiguous subsequence of σ which ends at x_i.

    $A[0] = 0$;
    $B[1] = x_1$
    $A[1] = max(0, x_i)$
    For $i = 2; i++; i <= n$
      $B[i] = x_i + max(0, B[i-1])$;
      $A[i] = max(B[i], A[i-1])$
    Write $A[n]$