CS 477/677 Study Guide for Examination October 23, 2024

`Sat Oct 19 10:52:11 AM PDT 2024`

1. Fill in the blanks.

   (a) Any comparison-based sorting algorithm on a file of size n must execute ____________ comparisons in the worst case. Use Ω.

   (b) The asymptotic time complexity of mergesort on an array of length n ____________. (Use Θ.)

   (c) The (worst case) asymptotic time complexity of treesort on n items is ____________.

   (d) The asymptotic time complexity of the Floyd Warshall algorithm on a weighted directed graph with n vertices and m arcs is ________________________. (Use Θ.)

   (e) The asymptotic time complexity of Dijkstra's algorithm on a weighted directed graph with n vertices and m arcs is ________________________. (Use O.)

   (f) The asymptotic time complexity of Treesort of n items is ________________________ in the worst case, but if you use a balancing scheme for the tree, you can expect that the asymptotic time complexity is ________________________.

   (g) ________________________ is a divide-and-conquer search algorithm which only works on a sorted list.

   (h) ________________________ is an $O(n)$-time search algorithm, generally used only when n is small.

2. Find an optimal prefix-free binary code for the following weighted alphabet.

a	18	
b	9	
c	3	
d	14	
e	23	
f	8	
g	7	

3. Given a binary search tree T which has n nodes and height h, what

   ○ $O(n)$
   ○ $O(h)$
   ○ $O(\log n)$
   ○ $O(\log h)$

4. Solve these recurrences

   (a) $F(n) = 2F(n/4) + \sqrt{n}$

   (b) $F(n) = 2F(n/4) + \sqrt{n}$

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

   (d) $F(n) = F(\log n) + 1$

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

   (e) $F(n) = F(\sqrt{n}) + 1$

   (f) $F(n) = 4F(\sqrt{n}) + 1$

   (g) $F(n) = 4F(\sqrt{n}) + \log^2 n$

6. Find the asymptotic time complexity of each of these code fragments in terms of n, using Θ notation.

   (a)
   ```
   for(int i = 0; i*i < n; i++)
   ```

   (b)
   ```
   for(int i = 0; i < n; i++)
    for(int j = 1; j < i; j = 2*j);
   ```

   (c)
   ```
   for(int i = 1; i < n; i++)
    for(int j = i; j < n; j = 2*j);
   ```

(d) `for(float x = n; x > 2.0; x = sqrt(x))` (`sqrt(x)` returns the square root of `x`.)

(e)
```
for(int i = 1; i < n; i = 2*i)
 for(int j = 2; j < i; j = j*j);
```

7. Write C++ code for the standard $O(n^2)$-time versions of selection sort and insertion sort, on an integer array A of size n.

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

void selectionsort()
 { // 4 lines deleted






 }

void insertionsort()
 { // 4 lines deleted






 }
```

8. In class, I presented three methods for solving the false overflow problem for the array implementation of queue, where items are inserted at one end and deleted from the other. There is a fourth method which is not available in every modern programming language; for example, it is not available in Pascal. What are those methods? (Do not give details. Just name each method in a word or a short phrase.)

9. Write pseudocode for an $O(n)$-time algorithm which, given a sequence $\sigma = (x_1, x_2, \ldots x_n)$ of positive numbers, computes the maximum sum of any subsequence of σ which contains no two consecutive terms of σ. For example, if $\sigma = (1, 4, 2, 1, 5, 3, 6, 7, 4)$, the maximum sum of such a subsequence is $4+5+6+4 = 19$.

10. Let $W_1 = 1$, $W_2 = 2$, and $W_n = 2W_{n-1} + 3W_{n-2}$ for $n \geq 2$. For example, $W_3 = 7$ and $W_4 = 20$. Find a constant K such that $W_n = \Theta(K^n)$.

11. The following function computes $x * n$. Find a loop invariant of the while loop.

```
float prod(float x, int n) // input condition: n >= 0
 {
  int m = n;
  float y = x;
  float z = 0.0;
  while(m > 0)
   {
    if(m%2) z = z+y;
    m = m/2;
    y = y+y;
   }
  return z;
 }
```

12. Write the prefix expression equivalent to the infix
epression $-a * b - (-c - d) \wedge e$
(Don't forget that $\wedge$ means exponentiation.)

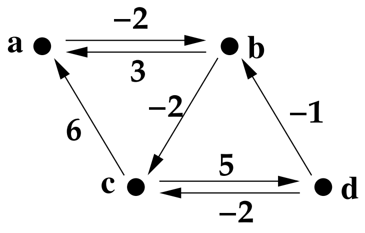
13. The following function computes x^n. Find a loop invariant of the while loop.

```
float pwr(float x, int n) // input condition: x > 0 and n >= 0
 {
  int m = n;
  float y = x;
  float z = 1.0;
  while(m > 0)
   {
    if(m%2) z = z*y;
    m = m/2;
    y = y*y;
   }
  return z;
 }
```

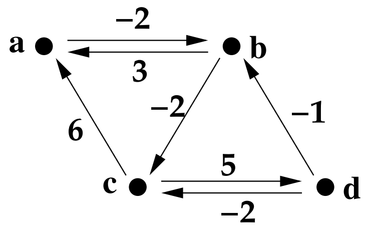
14. Write pseudocode for the Floyd Warshall algorithm on a weighted directed graph. Assume the vertices are the integers from 0 to $n-1$ and W$[i, j]$ is the weight of the arc from i to j. If there is no such arc, W$[i, j] = \infty$. Your code should calculate all V$[i, j]$, the smallest weight of any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

15. 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.

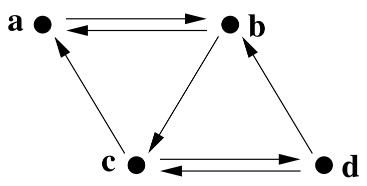
16. Consider the weighted directed graph shown below in (a). Since there are negative weight edges, the weights of the edges are adjusted, using the method given by Johnson's algorithm. Show the adjusted weights in (c). You can use (a) and (b) for your work.



(a)



(b)



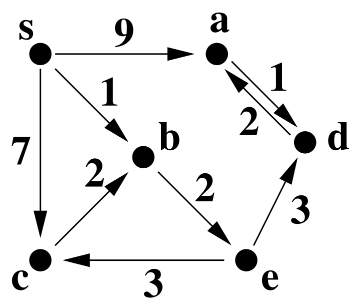
(c)

17. 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.

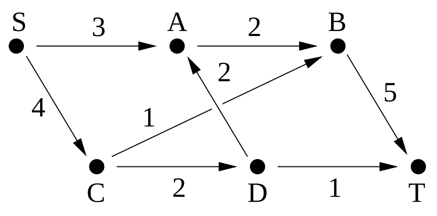
```
void swap(int&x,int&y)
 {
  int temp = x;
  x = y;
  y = temp;
 }

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?

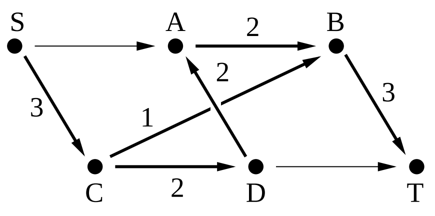
18. Use Dijkstra's algorithm to solve the single source minpath problem for the weighted digraph shown below.



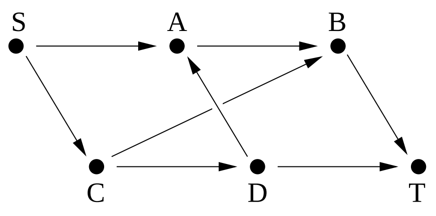
19. Find the maxflow and mincut in a weighted directed graph. Figure (a) shows the weighted digraph G. The goal is to maximize the flow from S to T. Figure (b) shows the initial flow. Draw the residual graph in Figure (c). Use the remaining space for your work. Be sure to indicate the mincut.



(a)



(b)



(c)