University of Nevada, Las Vegas Computer Science 477/677 Spring 2021

Practice Examination for February 9, 2021

The entire practice examination is 370 points.

1. True or False. [5 points each]

   (a) _____ Computers are so fast today that complexity theory is only of theoretical, but not practical, interest.

   (b) _____ If S is a set of distinct items, we say that an $x \in S$ has *rank* k if there are exactly k members of S which are less than or equal to x. If, while implementing quicksort to sort a set of n distinct items, if we always pick the pivot (cut) item to be an item whose rank is at least 10% of the size of the subset we are currently sorting, and never more than 90% of the size of that subset, the time complexity of our implementation will be $\Theta(n \log n)$.

   (c) _____ Any comparison-based sorting algorithm must use at least $\log_2(n!)$ comparison to sort n items, in the worst case.

2. Fill in the blanks. [5 points each blank.]

   (a) Name a well-known divide-and-conquer searchng algorithms.

   (b) Name three well-known quadratic time sorting algorithms.

   (c) Name two well-known $\Theta(n \log n)$ time divide-and-conquer sorting algorithms.

   ---

3. Solve the recurrences. Give asymptotic answers in terms of n. [10 points each.]

   (a) $F(n) = 4F(n/2) + n^2$.

   (b) $G(n) = G(n-1) + \log n$

(c) $H(n) = 2H(\sqrt{n}) + \log n$.

(d) $K(n) = K(n - \sqrt{n}) + 1$.

(e) $F(n) = 4F\left(\frac{3n}{4}\right) + n^5$ (No, you don't need a calculator.)

(f) $T(n) = T(n/2) + T(n/3) + T(n/6) + n$

(g) $T(n) = T(n/2) + T(n/3) + n$

(h) $T(n) = T(3n/5) + T(4n/5) + n^2$

(i) $T(n) = T(3n/5) + T(4n/5) + n$

(j) $T(n) = T(3n/5) + T(4n/5) + n^3$

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

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

(m) $F(n) = 3F(n/3) + 3F(2n/3) + n$

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

(o) This problem is a challenge. I have not been able to solve it. Can you?

$F(n) = F(n/2) + F(n-1) + 1$

4. [15 points] Consider the following procedure:

```
void george(int n)
 {
   int m = n;
   while (m > 1)
    {
      for (int i = 1; i < m; i++)
       cout << "I cannot tell a lie.  I chopped down the cherry tree." << endl;
      m = m/2;
    }
 }
```

Consider the question of how many lines of output the execution of `george(n)` would produce. Write down an appropriate recurrence for this question, and give an asymptotic solution in terms of `n`, using either O, Ω, or Θ, whichever is most appropriate.

5. [30 points] What follows is a portion of my C++ code of quicksort on an array of integers, with some parts deleted. Fill in the missing parts.

```
void quicksort(int first,int last) // sort A[first] ... A[last]
 {
  if(first < last) // otherwise it is already sorted
   {
    int mid = (first+last+1)/2;
    swap(A[first],A[mid]);  // Assume the swap procedure has been written
    int lo = first;
    int hi = last;
    int pivot =             // fill this in
    while(lo < hi)
                            // fill in lines here









                            // lo == hi at this point
    swap(A[first],A[hi]);   // place the pivot between the two subarrays
    quicksort(first,hi-1);  // recursive call
    quicksort(      ,     ) // another recursive call.  Fill in the parameters
   }
 }
```

6. [30 points] Walk through polyphase mergesort, where the input file is as given below.

MDYCOSEZVQWBXANLH

7. [30 points] Walk through mergesort, where the input file is as given below.

MDYCOSEZVQWBXANLH

8. [20 points] The following code implements an algorithm we've discussed in class, on an array A. What algorithm does the code implement?

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

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

9. [20 points] You are working on computer which lacks multiplication and addition. However, it can add or subtract 1 or 2. What does this function do?

```
int double(int n)
 // input condition: n >= 0
 {
  int p = n;
  int q = 0;
  while(p > 0)
   {
    p = p-1;
    q = q+2;
   }
  return q;
 }
```

10. [20 points]

What is the purpose of the following code?

```
float mystery(float x, int a)
 // input condition: a >= 0
 {
  if(a == 0) return 1.0;
  else if(a%2) // a is odd
   return x*mystery(x,a-1);
  else
   return mystery(x*x,a/2);
 }
```

11. [20 points]

Assume that $A[0] \ldots A[n-1]$ is a sorted array of integers, where n is a positive integer, and that b is an integer. The code below implements binary search, to decide whether h is an entry of the array. which is equal to b.

```
int lo = 0;
int hi = n;
```

```
while(lo < hi)
 {
  int mid = (lo+hi)/2; // truncated division, as in C++
  if(A[mid] < b) lo = mid+1;
  else hi = mid;
 }

if (           ) cout << "Yes" << endl;  // I need to insert a condition here!
else cout << "No" << endl;
```

What do you think the condition of the if statement should be?