University of Nevada, Las Vegas Computer Science 477/677 Fall 2022

Answers to Examination September 29, 2021

The entire examination is 240 points.

1. Fill in the blanks.

   (a) [10 points] Any comparison-based sorting algorithm on a file of size n must execute $\Omega(n \log n)$ comparisons in the worst case.

   (b) [10 points] Name two well-known divide-and-conquer sorting algorithms.

   quicksort

   mergesort

   (c) [10 points] The items in a priority queue represent

   unfulfilled obligations

   (d) [10 points]
   You have an array consisting of thousands of names in alphabetical order. What algorithm would you use to determine whether this array contains the name "Jackson Smathers"?

   binary search

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

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

   $\Theta(n \log n)$

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

   $\Theta(n)$

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

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

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

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

   $F(n) = \Theta(n)$

   (c) $F(n) = 2F(n/2) + n$ $F(n) = \Theta(n)$

   (d) $F(n) = F(3n/4) + F(n/2) + 3F(n/4) + n$

   $C = 1$ and $p = 2$ since $(3/4)^2 + (1/2)^2 + 3(1/4)^2 = 1$. Since $p > C$, The answer is $\Theta(n^p) = \Theta(n^2)$.

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

You simply have to remember this one. $F(n) = \Theta(\log^* n)$.

4. [10 points] The answer to each of these questions is either "bubblesort," "insertion sort," or "selection sort."

   (a) Treesort is a form of **insertion sort**.

   You insert items into a binary search tree one at a time, and after each insertion, the tree is in alphabetic order.

   (b) Heapsort is a form of **selection sort**

   Each round of heapsort selects the minimum (or maximum) remaining item.

5. [20 points] Consider the following procedure:

```
void F(int n)
 {
  if(n >= 1)
   {
    for (int i = 1; i < n; i++)
     cout << "hello" << endl;
    F(n/2);
    F(n/2+1);
   }
 }
```

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

The recurrence is $T(n) = n + 2T(n/2)$.

The solution is $T(n) = \Theta(n \log n)$.

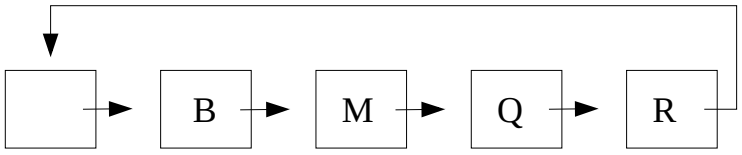
6. [20 points] Explain the implementation of a sparse array by using a search structure.

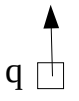
Let A be the sparse array. The search structure stores ordered pairs of the form (i, x), where $A[i] = x$. If there is no pair whose first term is i, then $A[i]$ is the default value.

To execute fetch(i), search the structure for an ordered pair of the form (i, x). If such a pair is found, return x. If none is found, return the default value.

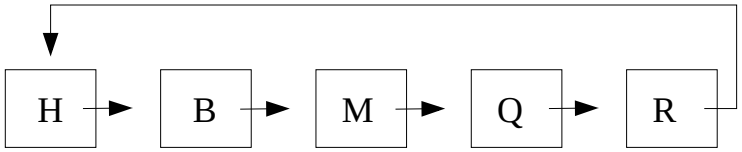
To execute store(i, y), search the structure for an ordered pair of the form (i, x). If one is found, overwrite x with y. Otherwise, insert the pair (i, y) into the structure.

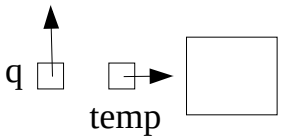
7. [20 points] Show a circular queue with dummy node items B, M, Q, R, in that order, from front to rear. then show how the queue changes when you insert H.



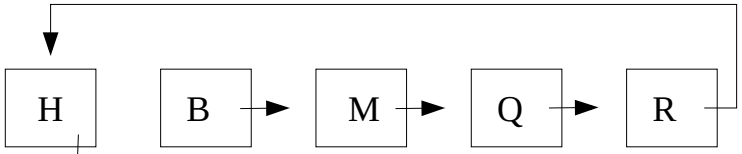


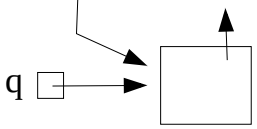
The initial queue. Static pointer q points to the dummy node.
Dummy points to front node. Rear node points to dummy.
All nodes are private; q is the only publically visible part of the queue.





New local variable temp points to a new node.
H, the new datum is written into the dummy node.





The pointer of the (old) dummy node is copied
to the pointer of the new node.
The value of temp is copied to the pointer q
The new node becomes the dummy node, and the old dummy is the rear node.
temp is deallocated. Static q is still the only public part of the structure..

8. [30 points] A stack of integers is implemented in C++ as a linked list as follows.

```
struct stack
 {
  int item;
  stack*link;
 };
```

Finish writing the code for the operators push, pop, and empty, below.

```
void push(stack*&s,int newitem)
 {
  stack*top = new stack;
  top->link = s;
  top->item = newitem;
  s = top;
 }

int pop(stack*&s)
 {
  int rslt = s->item;
  s = s->link;
  return rslt;
 }

bool empty(stack*s)
 {
  return s == NULL;
 }
```

9. [10 points] Let $F_1, F_2, \ldots$ be the Fibonacci numbers. Find a constant K such that $F_n = \Theta(K^n)$. Show the steps.

We assume $F_n = K^n$. (I have not presented a proof that this assumption is justified, but it is.)

Then

$$\begin{aligned} F_{n+2} &= F_{n+1} + F_n \\ K^{n+2} &= K^{n+1} + K^n \\ \text{divide by } K^n\text{:}\ \ K^2 &= K + 1 \\ \text{by the quadratic formula:}\ \ K &= \frac{1 \pm \sqrt{5}}{2} \\ \text{Fibonacci numbers are positive:}\ \ K &= \frac{1 + \sqrt{5}}{2} \end{aligned}$$

10. [20 points] Find the loop invariant of the following C++ function.

```
int sumdivisors(int n) // input condition: n > 0
 {
  int d = 1;
  int sum = 1;
  while(d < n)
   {
    d++;
    if(n%d == 0) sum = sum+d;
   }
  cout << sum << endl;
 }
```

The loop invariant is that the variable `sum` is the sum of all divisors of n in the range $1 \ldots d$. For example, if $n = 60$ and $d = 23$, then `sum` $= 1 + 2 + 3 + 4 + 5 + 6 + 10 + 12 + 15 + 20 = 78$