CSC 180, Exam III

Exam III Notes

- You may bring one page of notes (front and back) to the exam. This page may be handwritten or typed.
- Computer access will not be permitted during the exam.
- Cell phones must be put away at all times – but you may use calculators
- Don't hesitate to contact me if you have any questions!

Exam III Concepts

- Running time, Number of Operations, and Big Theta notation
  - $\theta(1)$ vs. $\theta(\log_2 n)$ vs $\theta(n)$ vs $\theta(n^2)$ vs $\theta(n!)$, etc
- Cleanup Algorithms and their running times
  - Shuffle-left
  - Copy-over
  - Converging pointers
- Searching Algorithms and their running times
  - Sequential Search
  - Binary search
- Sorting Algorithms and their running times
  - Select sort
  - Quicksort
- The Traveling Salesman Problem and it's running time
  - Fundamental Counting Rule
  - Heuristic for the Traveling Salesman problem and it's running time

Practice Problems

1. The algorithm below uses a *while* loop to count the number of times the number 7 appears in a list. Find the total number of operations (assignments, additions, comparisons, and print statements) that are executed and the order of magnitude (theta notation) for the running time under the following scenarios:

   a. The number 7 does *not* appear in the list
   b. The number 7 appears 1 time in the list
   c. Every element in the list is the number 7

```
n ← length of the list
index ← 0
count ← 0
while index < n :
      if mylist[index] == 7 :
            count ← count + 1
      index ← index + 1

Output the count
```

For questions 2 – 4, use the given *clean up* algorithm to clean up the following list: [1, 0, 2, 0, 4, 0]. Note that a 0 is an invalid value. Then repeat questions 2 – 4 for the list: [0, 3, 4, 0, 2, 6]

2. After each iteration of the Shuffle left algorithm, state the following:
    a. The value of *num_valid*
    b. The elements in the list

3. In the Copy-over algorithm, we first iterate through each element of the list and count the number of valid elements, which is *num_valid* = 3 for the first list. After each iteration of the second loop, state the elements that are in the copy list.

4. After each iteration of the *Converging pointers* algorithm, state the following:
    a. The values of *left* and *right*
    b. The value of *num_valid*
    c. The values in the list

5. Consider the list containing the integers: 1, 3, 4, 5, 7, 8, 9, 10, 11
    a. Show the binary search tree for this list
    b. What is the maximum number of comparisons that are needed to determine whether or not a number is in the list
    c. If sequential search were used, what would the maximum number of comparisons be?

For questions 6 – 7, use the specified *sorting* method to put the elements in sorted order (from smallest to largest): [19, 20, 3, 2, 6, 10, 9]. Then repeat these questions using the list: [13, 8, 4, 5, 9, 1]. Note: you should check your answers using the Practice Notebook under Sorting Algorithms in the Notes section.

6. Use *selection sort* to sort the given list. For each iteration of the loop, state the following:
    a. The maximum number found
    b. The two values that are swapped
    c. The updated list after the elements are swapped

7. Use *quicksort* to sort the given list. Create a binary tree that shows the following, similar to the trees in pages 16-17 of the notes:
    a. The list being looked at
    b. The list after all swaps from the *partition* algorithm

8. For the Traveling Salesman problem, how many ways are there to visit 6 different cities before returning home?

9. Using the nearest city heuristic for the Traveling Salesman problem, find the route to visit the cities labeled A – E in the image below. How many comparisons are needed for this heuristic?