CptS 223 – Advanced Data Structures in C++

Written Homework Assignment 1: Math Review, Big-O, Recursion and General Linux/Git Topics

Assigned: Monday, February 1, 2021
Due: Sunday, February 14, 2021

I. Problem Set:

1. (15, -1 pts/rank) Order the following set of functions by their growth rate (from fastest to slowest – rank 1 – 12, where 1 is the fastest and 12 is the slowest). Hint: you can plot their curves in a X-Y axis using http://fooplot.com/:

Unordered Complexities	Ordered Complexities
N	4
√N	2
N^1.5	7
N^2	8
N log N	5
N log(log(N))	3
N log^2 N	6
2/N	12
2^N	11
2^(N/2)	10
37	1
N^2 log(N)	9

2. (15 pts) A program takes 35 seconds for input size 20 (i.e., n=20). Ignoring the effect of constants, approximately how much time can the same program be expected to take if the input size is increased to 100 given the following runtime complexities?
    a. O(N)

t = time
t = kn
35 = k*20, t = k*100

t/35 = 35/20

t = 175 seconds

b. O(N + log N)
N dominates log N
O(N): **t = 175 seconds**

c. O(N^3)
t = kn^3
35 = k * 20^3, t = k * 100^3

t/35 = k * 100^3 / k * 20^3

t = 35 (100/20)^3

t = 4375 seconds

d. O(2^N)[1]
t = k * 2^n

35 = k * 2^20, t = k * 2^100
t/35 = k * 2^100 / k * 2^20

t = 35 * 2^80

4.23124 e 25 seconds

[1] You might need an online calculator with arbitrarily large numbers for this one. Scientific notation and 8 significant figures is just fine.

3. (10 pts) How many nodes in a complete trinary tree of depth 5? Hint: use geometric series.

S = a * (1-r^n/1-r)
= 1 * (1-3^5 / 1-3)
= 1* 121
= **121 nodes**

4. (15 pts) Write a simple recursive function to calculate (and return) the height of a general binary tree T. The height of a tree T is defined as the number of levels below the root. In other words, it is equal to the length of the longest path from the root (i.e., number of edges along the path from the root to the deepest leaf). Note that the term "nodes" is used to include both internal nodes and leaf nodes. You can assume the following tree node structure:

```
class Node
{
        Node *left; // points to the left subtree
        Node *right; // points to the right subtree
}

int get_height(Node* root)
{
        // if  root is null then the tree is empty
    if (root == nullptr)
    return 0;

    int height = max(get_height (root->left), get_height (root->right));
    return height;
}
```

5. (15 pts) Rewrite the pseudocode presented in class for the Fibonacci numbers *without* recursion (hint: use loop) and discuss the pros and cons of recursion compared to iteration.

Num = Fibonacci number Int x = 0, y = 0, z = 0 For x = 0; x < num; i++ z = x + y x = y y = z	Iteration is better because the function uses x and y to store the Fibonacci number. Storing the values in the function avoids using stack memory and reduces time complexity O(n) compared to recursion's O(2^n). The pros of a recursive solution simplicity. The function only takes up 3 lines of code and is much simpler.

6. (10 pts) What is Git and what is the purpose of using Git in general?

Git is a version control software that allows users to contribute changes to any public repository. Git is used to work on projects in a team setting and share changes between users. Instead of saving the code project on the local machine, git allows the user to upload to an online server where the project is stored online.

---

7. (10 pts) What is the Linux tool gdb? What is the difference between cmake and make?

gbd is the GNU project debugger used to see what the program is doing as it executes. You can chose stopping points for the debugger to stop at and move one line of code at a time.

Make is a software system that creates the executable files for the program to be carried out. The compiler compiles the code, then the Make tool creates the executable file to run the program.

CMake is the cross-platform version of Make. CMake recognizes the compiler used in the program to then create the executable files and run the program. CMake's advantages over Make are cross platform discovery of system libraries, automatic discovery of files, and easier compilation in shared libraries. Make is simpler can be used for smaller tasks that do not require as much complexion as CMake.

8. (10 pts) How do argc and argv variables get set if the program is called from the terminal and what values do they get set with?

```
int main(int argc, char* argv[])
{
        return(0);
```

```
}
```

Argc and argv variables are used for command line arguments from the command line in C and C++. The argc variable is argument count and argv is argument vector. argc contains the number of strings pointed to by argv. They can pass command line arguments from the terminal instead of running inside the program itself.

II. Submitting Written Homework Assignments:

1. On your local file system, create a new directory called HW1. Move your HW1.pdf file in to the directory. In your local Git repo, create a new branch called HW1. Add your HW1 directory to the branch, commit, and push to the remote origin which is your private GitHub repo.
2. Do not push new commits to the branch after you submit your link to Canvas otherwise it might be considered as late submission.
3. Submission: You must submit a URL link to the branch of your private GitHub repository. Please add the GitHub accounts of the instructor and two TAs (see Syllabus) as the collaborators of your repository. Otherwise, we won't be able to see your repository.

III. Grading Guidelines:

This assignment is worth 100 points. We will grade according to the following criteria:

- See above problems for individual point totals.