John O’Donnell
Dr Herve
CSC 412 Operating Systems
29 September 2020

Assignment 2 Report: Pointers, Arrays, and File System Navigation with Bash

prog02Part1.c

Sections 2 of the assignment description describes a program which takes an input set of three integers that describe a certain Fibonacci sequence. These are the first digit F1, the second digit F2, and the sequence’s desired length. The solution I implemented used only one data structure, an array of integers.

My solution deconstructed the problem into four basic functions. These basic functions are:

- validate if a given character array represents a valid integer,
- convert a given character string into an integer,
- generate a Fibonacci sequence from the input arguments,
- output the Fibonacci sequence to standard output.

The program uses an array allocated on the heap in order to store the digits of the Fibonacci sequence. It’s memory is allocated in the main function, and then a pointer to it is passed to the Fibonacci generator function along with the input arguments. As the function generates each number in the sequence, they are stored in the array in order. The array being allocated on the heap means that it can later be passed to the output function. The memory is freed after the output function is called.

Prog02Part2.c

Sections 3.1 and 3.2 describe a program which takes an arbitrary number of Fibonacci-describing integer sets and generates a number of Fibonacci sequences, all while tracking the unique values generated in all sequences.

My solution to this problem expanded on the my solution described for prog02Part1.c. I loop through the input triples, generating each Fibonacci on the heap. Pointer to these sequences are stored in an array on the heap, and the lengths of each sequence also occupy an allocated array. My solution adds two functions to those listed above:

- a function which, given a pointer to a Fibonacci sequence and a pointer to the array of unique values, determines which values in the sequence are not present in the unique array and appends them,
- a function which, given a pointer to an array of pointers to Fibonacci sequences, a pointer to an array storing the lengths of those sequences, and the total number of sequences, outputs each sequence in turn,
- a function to output the array of unique values,
- a function that frees all allocated memory pointed to by an array of pointers.

The program uses heap-allocated arrays in a few ways. First, memory is allocated for two arrays: one to hold pointers to all the Fibonacci sequences, and another to hold every unique value generated over the programs lifetime. Then the input arguments are parsed, and each time a Fibonacci sequence is

generated, memory is allocated for it on the heap, and its pointer is stored in the previously mentioned array. I designed my program this way because my program was outputting the first Fibonacci sequence even if the second triplet of input was invalid. This way, all the sequences have to been validated and generated before any output is made.

I implemented everything in the program description correctly. I had a few bugs along the way, early on I forgot to ensure that F1 was greater than 0. However, the program in it's current iteration is pretty robust. I haven't thought of an input that is able to break it.