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, adding two functions:

- 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 to output the array of unique values.

The program uses heap-allocated arrays in two ways. The array for unique values is allocated near the beginning of the main function. The main function contains a for loop, which validates and converts the next set of Fibonacci-defining arguments and generates the sequence in question. Each iteration of the loop allocates a new array and fills it with the elements of the new sequence. At this point, the Fibonacci array and the unique value array are passed to the unique value scraping function. Upon completion, the Fibonacci array is output, and the freed. The unique value array is freed at the end of main.