Chirag Chadha
PA 4

In PA4, I created an assembler that would read Y86 code and compile and run the instruction on an Intel X86 machine. This program consisted of many functions, which accounted for every kind of instruction. The first thing I made my assembler do is to read the text file and separate the directive, location address, and instruction. The very first line consisted of the size directive which indicated the memory needed to be allocated for the program. I converted the hex-size to a decimal value and created a pointer with a malloc of the given size. Then I stored the instructions in memory. I separated the directive, memory address, and instruction. I stored the given instruction in its respective memory location. Each instruction was stored in a particular way depending on its directive. I programmed every instruction and manipulated what each register did in assembly. This program was very challenging in the beginning because there were multiple factors to consider when creating the assembler. The first problem I was having was properly allocating memory for the instructions. My instructions were going in random locations in my array. I realize that it was a logical error with the algorithm and corrected it. However after concluding the project everything has been working. The project stressed the core fundamentals behind registers and how an emulator functions. This project was very hard to complete at first, however as I continued to learn through the process, I began to develop a grasp on how the program was supposed to operate. For my disassembler, I evaluated the hex addresses stored in my memory array to determine which kind of instruction it was. I returned the instruction in terminal and provided a list of destination registers to go with it. The hardest part of this assignment for me was to picture what was going on. I began by drawing tons of pictures to understand how the memory was being allocated and how the process of an emulator and disassembler worked.