IBM 650 Emulator in Python Design
Topic Paper #4

Alex Laird
CS-3210-01
1/30/09
Survey of Programming Languages
Spring 2009
Computer Science, (319) 360-8771
alexdlaird@cedarville.edu

Grading Rubric		Max	Earn
	On Time/Format	1	
	Correct	5	
	Clear	2	
	Concise	2	
	Total	10 pts	

Peer Reviewer

ABSTRACT

This paper talks about my intended implementation design for an IBM 650 emulator written in the Python programming language.

Keywords

IBM 650, Emulator, Python, Design

1. INTRODUCTION

Essentially, the emulator made will read an input stream (either from the command line or from a specified file), parse the file into organized commands (one command per line), execute each individual command, and then produce the output (either to the shell or to a specified file).

2. READING AND PARSING THE FILE

The file is read in line-by-line. Each line is parsed into individual statements and executed. Each file has three sections: initial data values, program instructions, and input data. Each section is separated by +9999999999.

Each individual command is parsed into four executable parts. The first +# or -# is the operation to be performed. The next three digits is the memory location of the first operand. The next three digits is the memory location of the second operand. The final three digits is the memory location of the destination.

3. EXECUTION OF STATEMENTS

The execution functions are all elif statements reading the parsed values from first symbol and digit of input; depending on what the values are, different commands are executed. A list of commands and their operations is shown in Figure 1.4 of McClenan IBM 650 PDF file. The execution is then done to the three provided memory locations.

What would be memory locations on the IBM 650 will be represented by an array in the Python emulator. When the entire list of operations has been completed, the array can simply be output in order.

4. OUTPUTTING RESULT

The output will be either to the shell, if no output file is specified, or to a text file. A simple loop will run through the storage array to print each value.

5. POTENTIAL PROBLEMS

Deciding whether to read in each line as an integer or as a string may present problems. It will probably be easiest to read the lines in as strings and parse them that way, then converting each parsed set to integers.

6. CONCLUSIONS

The IBM 650 emulator will be a fairly simple application written using Python. The brunt of the application work will be done in a large set of elif statements. Simple read/write will be performed on the intput/output statements, and data storage will be simple enough into an array. In general, the emulator should be pretty simple with only a few minor implementation roadblocks.

Alex Laird, Cedarville University, Cedarville, Ohio, 45314