Birzeit University

Department of Electrical & Computer Engineering

Second Semester, 2020/2021

ENCS313 Linux Laboratory

Shell Scripting Project – Stack interpreter

Problem:

You are required to build an interpreter for a machine that has a single stack. The machine is primitive and thus it understands simple commands. Consider the following very primitive language for programming a stack machine:

Command	Meaning
int	push the integer *int* on the stack
+	push a '+' on the stack
s	push an 's' on the stack
e	evaluate the top of the stack (see below)
p	print content of the stack
d	delete the top of the stack
x	stop (exit the program)

Below is a brief description for each command in addition to an example in each case (the symbol > in the below examples refers to the prompt where the interpreter receives the commands). Note that the system must print the contents of the stack including the last input before executing the last input/command.

- **p: print content of the stack.**
  Example:
  Stack before: + 1 2 5 s
  > p
  p + 1 2 5 s // printing the contents of the stack including the last input/command
  + 1 2 5 s

- **int: push the integer int on the stack.**
  Example
  > 110
  The above will push the integer 110 on top of the stack.

- **e: evaluate or execute a command. It's behavior depends on the top of the stack:**
  - If **+** is on top of the stack, then the + is popped off the stack, the next two integers are popped and added, and the result is pushed back on the stack.
  - If **s** is on top of the stack, then **s** is popped off the stack and the next two items are swapped on the stack (thus the 2 elements remain on the stack).
  - If d is on top of the stack, then d is popped off the stack and the current top of the stack is removed from the stack.
  - If an integer is on top of the stack or the stack is empty, the stack is left unchanged.

In addition to the above commands, the system support the undo command (only when the input is from the terminal). Therefore, when the user enter u the system must return the contents of the stack before the execution of the last command. For example:

Stack before: + 1 2 5 s
> e
e + 1 2 5 s // first: printing the contents of the stack including the last input/command
3 5 s // second: printing the contents after executing the last input/command
> u
u 3 5 s // first: printing the contents of the stack including the last input/command
+ 1 2 5 s // second: printing the contents of the stack including the last input/command
>

The following examples show the effect of the e command in various situations; the top of the stack is on the left:

Stack before	Stack after
+ 1 2 5 s ...	3 5 s ...
s 1 + + 99 ...	+ 1 + 99 ...
1 + 3 ...	1 + 3 ...
d 1 2 5 s ...	2 5 s ...

The input to the program is a series of commands, one command per line as shown above. Your interpreter should prompt for commands with the symbol >. Also, the program can read the series of commands from a file. Assume that the stack deals only with unsigned integer numbers. Assume as well that the only allowed arithmetic command is +. In addition, assume that the allowed logical commands are & (AND), | (OR) and ^ (XOR). The interpreter should be able to handle errors if encountered. An example of an error you might get is when you're adding 2 popped elements from the stack, but one of the 2 elements is not an integer (e.g. + or &).

Submission:

Please submit the following:

1. Shell script program

2. Report: the report must include:
   a. The code, idea, and a screen shot of each task or stage.
   b. At least 2 testing examples.

Notes:

- Write the code for the shell script to satisfy the requirements described above and name the script as SimpleStack.
- Make sure your code is clean and well indented; variables have meaningful names, etc.
- Make sure your script has enough comments inserted to add clarity.
- Work in groups of at most two students
- Deadline: 22 April, 2021 at 11:59pm. Please submit your project (code + report) through ITC.
- This project is per group effort: instances of cheating will result in you failing the lab.