Assignment I
Due Friday, January 14, 2005 at 11:59pm

Programming Part

This part of assignment asks you to write a short Cool program. The purpose is to acquaint you with the Cool language and to give you experience with some of the tools used in the course.

A machine with only a single stack for storage is a *stack machine*. 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)
d	display contents of the stack
x	stop

The 'd' command simply prints out the contents of the stack, one element per line, beginning with the top of the stack. The behavior of the 'e' command depends on the contents of the stack when 'e' is issued:

- If '+' is on the top of the stack, then the '+' is popped off the stack, the following two integers are popped and added, and the result is pushed back on the stack.
- If 's' is on top of the stack, then the 's' is popped and the following two items are swapped on the stack.
- If an integer is on top of the stack or the stack is empty, the stack is left unchanged.

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*
$+ 1\ 2\ 5\ s \ldots$	$3\ 5\ s \ldots$
$s\ 1\ +\ +\ 99 \ldots$	$+\ 1\ +\ 99$
$1\ +\ 3 \ldots$	$1\ +\ 3 \ldots$

You are to implement an interpreter for this language in Cool. Input to the program is a series of commands, one command per line. Your interpreter should prompt for commands with `>`. Your program need not do any error checking: you may assume that all commands are valid and that the appropriate number and type of arguments are on the stack for evaluation. You may also assume that the input integers are unsigned.

You are free to implement this program in any style you choose. However, in preparation for building a Cool compiler, we recommend that you try to develop an object-oriented solution. If you find any code in `~c22/cool_project/examples` that you think would be useful, you are free to use it.

Sample session. The following is a sample compile and run of our solution.

```
%coolc stack.cl atoi.cl
%spim -file stack.s
SPIM Version 5.6 of January 18, 1995
Copyright 1990-1994 by James R. Larus (larus@cs.wisc.edu).
All Rights Reserved.
See the file README a full copyright notice.
Loaded: /home/c22/cool_project/lib/trap.handler
>1
>+
>2
>s
>d
s
2
+
1
>e
>e
>d
3
>x
COOL program successfully executed
```

Getting the assignment.

Use your CS id and password to log on to one of the T-lab machines in the front room of the lab. If you do not yet have a keycard for the lab, you may get one from the department admistrator, Karen Healy Stover. If you ssh from another location, the host name is `tlab-xx.cs.northwestern.edu` where `xx` is a number between 10 and 14.

Create a working directory called PA1 and `cd` into it. From there, type

```
gmake -f ~c22/cool_project/assignments/PA1/Makefile
```

This command creates several files you will need in the directory. Follow the directions in the README file. The README file explains how to turn in your work when you are finished and contains an additional question that you need to answer as part of the assignment.

Written Part

This part of the assignment asks you to solve some fairly straighforward problems involving regular expressions. Try to come up with answers that are as short as possible.

1. Write a regular expression for the language of all strings on alphabet $\Sigma = \{a,\ b\}$ that do not contain more than two bs in a row.

2. Write a regular expression for the language of all strings on alphabet $\Sigma = \{a,\ b,\ c\}$ whose length is odd.

3. Define a DFA (Σ, Q, s_0, δ, F) that recognizes all strings of as and bs that do not contain the subsequence (*not* substring) abb. Try to make it as small as possible. Given the diagram, come up with an equivalent regular expression. (Hint: start by drawing the state transition diagram of such a DFA. What would each state represent?)

Turning in the written part

Write your answers in a **text** file and email it to `c22@cs.northwestern.edu` along with the programming part.