CSCI 4160 Project1

Due: see class calendar

Goal:
To be familiar with COOL: classroom object-oriented language. Please read chapters 3-11 in the manual before working on the project.

Description:
This assignment asks you to write a short Cool program. The purpose is to acquaint you with the Cool language.

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
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 multiplied, and the result is pushed back 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*
* 10 2 5 . . .	20 5 . . .
11 * 33 . . .	11 * 33 . . .

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. Your interpreter should exit gracefully; do not call abort() after receiving an x.

You are free to implement this program in any style you choose. If you wish, you may copy A2I class defined in atoi.cl and List and Cons classes in list.cl to your solution file so that you can use them to perform string to integer conversion and manipulate lists. Both files could be found in example folders in class repository.

Sample session

The following is a sample execution of your solution. To run the COOL interpreter, open a command line window and type the following command:

%cool_windows solution.cl

```
>15
>2
>*
>d
*
2
15
>e
>e
>d
30
>x
```

COOL program successfully executed.

If you don't have a windows machine, you can always use the department jupyterhub server. Here is a short description of using jupyterhub server:

1. login to jupyterhub.
   https://jupyterhub.cs.mtsu.edu/azuread/
2. Configure git with your email and username.
   git config --global user.email "Your MTMail"
   git config --global user.name "Your name"
3. Clone class repository in the CURRENT folder:
   git clone https://gitfront.io/r/zdong-mtsu/cxBi197e9nLJ/CSCI-4160.git
4. Copy project files to a different location and start working there.
5. Under the folder that contains your project file and COOL interpreter, type the following command to make the interpreter executable: *chmod a+x cool_Linux64*
6. to run your solution, use the following command: ./cool_Linux64 solution.cl

How to submit: Submit your source file to D2L dropbox of project1.