# Bonus HW: Implement RBMRCFAE(supporting laziness) parser and interpreter by using your favorite programming language

**\* This bonus task must be done by yourself. Discussion with your classmates is allowed! If you have any issues, discuss with JC and TAs.**

If you can complete this Bonus HW. Your final grade will be updated as follows:

| Original grade |  | Final grade |
| --- | --- | --- |
| F | → | B |
| D and D+ | → | B+ |
| C to B | → | A |
| B+ and A | → | A+ |

This task is to implement RBMRCFAE (with laziness) in your favorite programming language (except for Racket). The parser and interpreters must support command line interface (CLI). For example (java example),

| $> java -cp [class\_path\_settings] [your\_class\_name\_that\_has\_main\_method] "{+ 2 3}"  5 |
| --- |

Your program must get the '-p' option that only enables a parser. For example,

| $> java -cp [class\_path\_settings] [your\_class\_name\_that\_has\_main\_method] -p "{+2 3}"  (add (num 2) (num 3)) |
| --- |

In case you are using JavaScript, you can create a web page with a simple input box that can have text code as input.

\* **RBMRCFAE** : Support boxes and variables for RCFAE with Laziness. Both call-by-reference and call-by-value must be implemented as learned. It is recommended to follow the algorithm we learned in lectures. That could be an easier way to achieve the final goal. In case you think you are smart, then you can try new ideas and algorithms.

* Example Input 1 (you can ignore new lines for your input.)  
  "{with {swap {refun {x}  
   {refun {y}  
   {with {z x}  
   {seqn {setvar x y}  
   {setvar y z}}}}}}  
   {with {a 10}  
   {with {b 20}  
   {seqn {{swap a} b}  
   b}}}}"
  + Output 1  
    (v\*s (numV 10) (aSto 3 (numV 10) (aSto 2 (numV 20) (aSto 4 (numV 10) (aSto 3 (numV 20) (aSto 2 (numV 10) (aSto 1 (refclosV 'x (refun 'y (app (fun 'z (seqn (setvar 'x (id 'y)) (setvar 'y (id 'z)))) (id 'x))) (mtSub)) (mtSto)))))))))
* Example Input 2  
  "{with {swap {refun {x}  
   {refun {y}  
   {with {z x}  
   {seqn {setvar x y}  
   {setvar y z}}}}}}  
   {with {a 10}  
   {with {b 20}  
   {seqn {{swap a} b}  
   a}}}}"
  + Output 2  
    (v\*s (numV 20) (aSto 3 (numV 10) (aSto 2 (numV 20) (aSto 4 (numV 10) (aSto 3 (numV 20) (aSto 2 (numV 10) (aSto 1 (refclosV 'x (refun 'y (app (fun 'z (seqn (setvar 'x (id y)) (setvar 'y (id 'z)))) (id 'x))) (mtSub)) (mtSto))))))))

## 1. Due date

* Code: 10PM December 23 (Fri) 2022.
* Oral interview: Make an appointment (Offline or Zoom meeting) when you submit the code. Oral interview must be done before/on Dec 29 (Thu).
  + If you finish early, you can also have a meeting early.

## 2. What to submit

* Create a zip file, BonusHW-[Student\_id]\_[name].zip (which contains a readme file and all your code).
  + Name your files
    - BonusHW-README (any text/document types are fine)
    - BonusHW-[Student\_id]\_[name]\_code.zip
* Then, upload your zip file via LMS and send me an email. (You must send me an email after you upload your file in LMS. Also put your preferred time for the oral interview.)
  + JC's calendar: <https://lifove.github.io/calendar.html>
* No late submission
* No partial points.

\* You can discuss with your classmates to solve the problem. But all the code must be written by yourself. Start with the starter package: <https://drive.google.com/drive/u/0/folders/1li_NtFeOb3BUXH3LWM28n6y0PVcQl7dt>