# Homework: VarLang

**Learning Objectives:**

1. Get familiar with the concepts of scope, free and bound variables and environment
2. Write programs in VarLang
3. Understand and extend VarLang interpreter

**Instructions:**

1. Total points: 57 pt
2. Early deadline: Feb 16 (Wed) 11:59 pm, Regular deadline Feb 18 (Fri) 11:59 pm.
3. Download hw3code.zip from Canvas
4. Set up the programming project following the instructions in the tutorial from hw2 (similar steps)
5. How to submit:
   - For questions 1–3, you can write your solutions in latex or word and then convert it to pdf. Please provide the solutions in one pdf file.
   - For questions 4–6, please submit your solutions in one zip file with all the source code files (just zip the complete project's folder).
   - Submit the zip file and one pdf file to Canvas under Assignments, Homework 3

## Questions:

1. (3 pt) Write Varlang programs:
   (a) (1 pt) Convert 15 miles to kilometers using 1 mile = 1.61 km.
   (b) (2 pt) Write a VarLang program that evaluates to value 312. The program must include at least 2 let expressions and there must be a "hole in one of the scopes."(The definition of this kind is seen in section 3.3 of Rajan-PL book pg 48), For example:
   (let ((x 2) (y 4)) (let ((x 5) (z 10)) (let ((z 20)) (- (* x y) z))))
   $ 0
2. (6 pt) Compute the values of the following Varlang expressions. (Return an error if the values cannot be computed.) Please list the steps to show that you understand the scoping rules and semantics of the VarLang program.
   *Hint: section 3.5 of Rajan-PL book*
   (a) (2 pt) (let ((c 3)(d 4)) (+ c d))
   (b) (2 pt) (let ((a 5) (b 3)) (let ((b 5))(* b a)))

(c) (2 pt) (+ (let ((d 7)) d) (let ((c 2))(* c d)))

3. (8 pt) List free and bound variables for the following Varlang expressions:

   (a) (2 pt) (let ((q 5)) (let ((r q)) s))
   (b) (2 pt) (let ((c 3)(l p))(let ((s 2))(+ c l s)))
   (c) (2 pt) (let ((x 1)(y 1))(+ x y z))
   (d) (2 pt) (let ((q 5)) (let ((r q)) r))

4. (10 pt) This question's goal is to help you understand the idea of initial environments in programming languages. Extend the Varlang interpreter such that Varlang starts with an environment which predefines the ASCII values for characters from 'a' to 'z'.

   For example:

```
$ a
97
$ b
98
```

5. (15 pt) Extend the interpreter: Notice from the semantics and from the implementation of the let expression that Varlang doesn't place any restriction on defining the same variable two or more times in the same let expression. In fact, a variable can be defined any number of times and only the rightmost definition would have effect in the body of the let expression. So the program (let ((a 3) (a 4) (a 2) (a 342)) a) would give the answer 342. Modify the semantics of the Varlang programming language so that all variable names defined in a let expression must be unique. Otherwise, throw error.

   For example,

```
$ (letunique ((a 3) (b 4)) (+ a b))
7
$ (letunique ((a 3) (a 4)) a)
error
$ (letunique ((a 3) (a 4) (a 2) (a 342)) a)
error
```

   You will need to modify the following files-

   - (4 pt) Grammar file (Varlang.g)
   - (3 pt) AST file (AST.java)
   - (2 pt) Printer file (Printer.java)
   - (6 pt) Evaluator (Evaluator.java)

6. (15 pt) Extend the interpreter: Security is a major concern for any system. To deal with this it becomes important that deallocated memory of some program does not contain the data which can be read by malicious programs thereby causing leak of information. To avoid such information leak due to environment storage, we can augment the Varlang language with a *lete* (encoded let) expression that encodes the value before storing in environment and *dec* expression that decodes it prior to using it. Extend the Varlang programming language to support these two expressions. Implement an

encrypted let (lete for let encrypted), which is similar to let but it uses a key and a dec expression that is similar to VarExp. All values are stored by encrypting them with key, and read by decrypting them with key.

For example,

```
$ (lete 2 ((x 1)) x)
3
$ (lete 20 ((x 1)) (dec 20 x))
1
$ (lete 10 ((y 8)) y)
18
$ (lete 10 ((y 12)) (dec 10 y))
12
```

You will need to modify the following files-

- (4 pt) Grammar file (Varlang.g)
- (3 pt) AST file (AST.java)
- (2 pt) Printer file (Printer.java)
- (6 pt) Evaluator (Evaluator.java)