Problem Set 8
Comp 301
Fall 2021
Week 10: 06.12.2021 - 10.12.2021

Instructions:

- Submit your answers to the Blackboard PS8 assignment until December 11th Saturday, at 23.59.
- Please use the code boilerplate, which includes several tests for you to see if your code is correct.
- Submit your code and PDF file to BlackBoard as a single zip file *yourIDno_ username.zip*. (Example: *123456_ akutuk21.zip*)

Problem 1. : Write an `unparse` function, which works similar to `unparse-lc-exp` but now, it handles with the cases of `PROC` language. It takes an abstract syntax tree and build the expression. Below is an example:

```
> (unparse-proc
   (scan&parse "let x = 10
                in if zero?(x) then
                -(x, 10) else
                let y = 20
                in proc(a) -(a, -(x,y))"))

"let x = 10 in if zero?(x) then -(x,10) else let y = 20 in proc(a)
   -(a,-(x,y))"
```

Hint: You need to make changes in `interp.scm`
Hint: You need to use `string-append` function to append a list of string into a single string. You can check its usage here

Problem 2. : Write an `unparse` function, which is similar to what you have done in Problem 1, but now it handles expressions of LEXADDR language. So, instead of a regular abstract syntax tree, you need to handle translated abstract syntax tree.

```
> (unparse-nameless
   (translate "let x = 10
               in if zero?(x) then
               -(x, 10) else
               let y = 20 in proc(a) -(a, -(x,y))"))

"lexlet 10 in if zero?(#0) then -(#0,10) else lexlet 20 in lexproc
   -(#0,-(#2,#1))"
```

Hint: Check `lang.scm` and see which expressions are different than `PROC` language. Then, you make changes in `interp.scm`
Note: You are provided a function `translate` under `top.scm`. You can use it to discover how a translated AST looks like. Use the inputs from test cases and observe their ASTs.