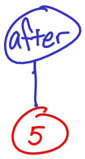
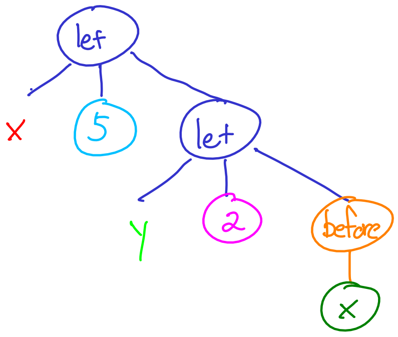
Compilation

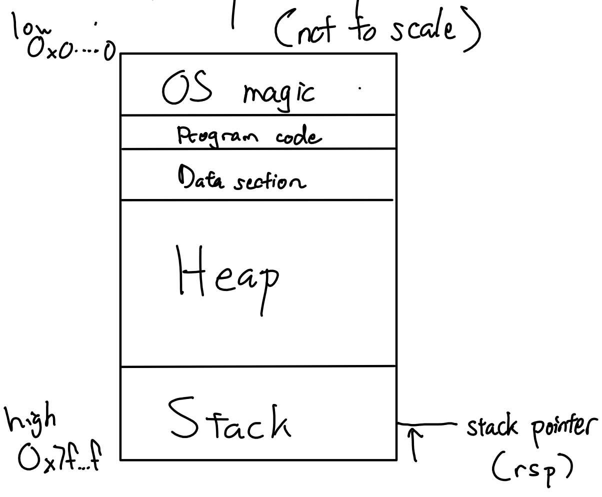
Translation of an expression in Auklet to assembly. The result described by the expression must be stored in rax by the end of executing the assembly.

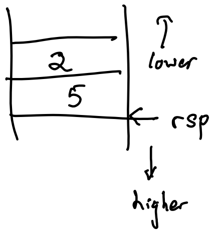
```
<expr> ::= 0 | 1 | -1 | 2 | -2 | ....
         | <expr> + <expr> | ....
         | after (<expr>) | before (<expr>)
         | let <var> = <expr> in <expr> | <var>
```

Auklet Code	AST	Assembly	
4		mov rax, 4	mov $4, %rax
after (5)		mov rax, 5 add rax, 1	
let x = 5 in let y = 2 in before (x)		mov rax, 5 mov [rsp-8], rax mov rax, 2 mov [rsp-16], rax mov rax, [rsp-8] sub rax, 1	mov $rax, -8(rsp)

Memory Layout

(not to scale)





```
let rec compile (env:enviorment) (e : expr) : instruction list =        RAX contains
                                                                          answer!
    match e with
    | EInt n -> [AsmMov (ArgRegister RAX, ArgConstant (string_of_int n))]
        [mov rax, (string_of_int n)]
    | EAfter (e') ->
        compile env e' @ [add rax, 1]
    | EVar (varname) ->

                              defn   body
                                ↓      ↓
     | ELet (varname, e1, e2)
```

type environment =
int * argument Map.String.t

$$\left(24 , \left\{ \begin{array}{l} x \mapsto [rsp-8], \\ y \mapsto [rsp-16] \end{array} \right\} \right)$$

C++
pair<int, Dictionary(string, argument)*>