

---

**TD6 - Friday, October 29**


---

**Exercise 1** (SSA form - Finish the exercise from previous TD)

Suppose we have the following code.

```

a := 200
b := a - 2;
i := 0;
L : if (a > i) goto E
b := b - 2;
i := i + 3;
c := a + b;
goto L
E : g := 2 * b

```

– Establish a connection between the dominance frontiers of the variables and the place where the  $\phi$  functions were added. Apply the Iterated Dominance Frontier Algorithm (seen in course) for passing the code to SSA form.

– The  $\phi$  functions are not machine instructions, obviously. So, before translating a program in SSA to machine code, we have to replace these functions but to keep the same semantics. Propose a solution. What problems appear? What solutions can we find to them?

– Give the conversion out of the SSA form for this program.

**Exercise 2** (Procedures and Functions - Escape Analysis)

We divide the local variables of a function  $f$  in two groups :

– variables used only by  $f$  - if there is enough space, these variables can be allocated in registers  
 – variables used outside  $f$  - for example, a variable accessed by a procedure nested inside the current one.  
 (Give the other cases when a variable is used outside  $f$ ).

We say that these variables "escape", and we are forced to allocate them in the stack frame.

Consider the following code. What are the variables that "escape"? What are the variables that are kept in registers and why?

```

int f(int a, int b)
{
  int c[3], d, e;
  d=a+1;
  e=g(c,&b);
  return e+c[1]+b;
}

```

Give a simple algorithm for finding "escape" variables (hint : use the AST). Should this algorithm be run before or after the semantic analysis?