Compilators, Assignment 5 theory

Eirik Jakobsen, Arve Nygård

12.03.2014

Problem 1, Type checking

We could (should) do type checking on return types for function - they should match the declared return type. Conditionals should be bool (i.e. `if`)

Problem 2, Stack

```
main()
Local variable
Local variable
Parameter 1
Return address
Saved FP
Local variable
Local variable
Local variable
Parameter 1
Parameter 2
Parameter 3
Return Address
Saved FP       <-- FP
Local variable
Local variable <-- SP
```

Problem 3, Assembly Programming

Source code

```
INT FUNC f(INT a, INT b) START
    INT c;
    c := a;
    RETURN g(b,c);
END
INT FUNC g(INT a, INT b) START
    RETURN b;
END
```

Corresponding Assembly

```
f:
    PUSH lr             ; store return address
    PUSH fp             ; store old fp
    MOV fp, sp          ; update fp

   PUSH r1           ; Declaration of local variable. Push to stack. Shuold get address fp+4.

                        ; >c:=a:
    PUSH [fp, #12]      ; push a (fp+12 is first parameter)
    POP r1              ; r1 contains value of a
    STR r1, [fp, #-4]   ; store r1 into c.

                        ; >Starting call to g()
    PUSH r1             ; Store registers on stack
    PUSH [fp, #8]       ; push parameters
    PUSH [fp, #-4]
    BL g                ; jump to g and store return address

                        ; >G returned.
    POP                 ; Clear parameters
    POP

    POP r1              ; Restore registers

    MOV sp, fp          ; clear locals.
    POP fp              ; restore fp
    POP lr              ; restore lr
    B lr                ; r0 already contains correct value, jump to caller.


g:
    PUSH lr             ; store return address
    PUSH fp             ; store old fp
    MOV fp, sp          ; update fp
    LDR r0, [fp, #8]    ; start return
    MOV sp, fp          ; clear locals.
    POP fp              ; restore fp
    POP lr              ; restore lr
    B lr                ; jump to caller
```