

CS 415: Compilers - Problem Set 4

Due: Monday, May 8

Problem 1

Suppose you need to implement the following routine

```
#define NUM_REGISTERS n
void getReg(STEntry *e, int *reg, boolean *needToLoad)
```

where e is the symbol table entry for some identifier (variable). Assume that, when generating the intermediate code (e.g., three address code), all temporary names and their types are added to the symbol table the first time they are used. `getReg` must return a register that can be used to hold the value of the variable in reg . `needToLoad` is true if the value of the variable is already in reg and false otherwise. reg should always be in $[1, n-1]$.

1. Sketch out the data structures and algorithm that you would use to implement `getReg` assuming you only have to deal with variables of basic types (int and character). (Hint: what do you have to do if there's not enough registers to hold everything you want/need?)
2. What complications are introduced when you have to handle arrays of basic types? How might you handle this?

Problem 2

For the following program

```
program Bogus;
  var a, b, c, d: integer;
begin
  a := 0;
  b := 1;
  c := 2;
  d := (a + b) * c
end.
```

1. Generate the corresponding abstract syntax tree.
2. Generate the corresponding three address code.
3. Generate Spim code, including data layout, using your `getReg()` from Problem 1, assuming that $n = 3$ (meaning you have 2 registers to play with). Comment your assembly code!

Problem 3

Generate three address code and then Spim code, including data layout, for the following program. For this problem, you can assume that you have all Spim registers. Again, make sure to comment your assembly code!

```
program Bogus;
  var a: integer;
  procedure moreBogus (var X: integer);
    var b: char;
        c: integer;
        d: array [0..4] of integer;
  begin
    d[2] = 5;
    X = d[2]
  end;
begin
  moreBogus(a)
end.
```