McMaster University Comp Sci 4TB3/6TB3, Winter Term 2018/19 — Lab 10
For the Labs on March 20 - March 23,
Due Monday, March 26, 11 pm

Eden Burton, Jenny Wang, Spencer Park

out of 20 points

- *Submission is to be done exclusively through Avenue. Submissions via e-mail will **not** be accepted. A **10% penalty** will be accessed for each day the lab is submitted after the due date*.
- You need to download the P0 compiler posted together with **lab 9**.
- This assignment requires access to a Linux, MacOS X, or some other Unix computer. You can log in remotely to either `moore.mcmaster.ca` or to `mills.mcmaster.ca` with ssh. Submissions are tested on `moore.mcmaster.ca`, please check if your submission works there.
- In this lab, you are allowed to work in pairs, provided that you split the work equally and arrive at a common understanding of the solution. However, in that case you must state in your submission the person you worked with, such that similarities in the solution will not be construed as Academic Dishonesty. Working in groups of three or larger is not allowed and will be considered Academic Dishonesty. If you look for someone to work with, we will try to find a match, please contact the TAs.
- You are allowed and encouraged to talk to everyone in the course to get a common understanding of the problem, but you can share partial solutions only with your collaborator, if you work in a pair. The final submission must be your own, that is, you cannot submit identical submissions with two names on them.
- In the lab sessions, the solution to last week's lab questions are discussed and you can get help with this week's lab questions. Attendance at the labs is not checked.

The P0 language supports value and reference parameters. The P0 compiler passes all parameters on the stack. On the MIPS processor, by convention, registers $a0 to $a3 are used for the first four parameters, provided they fit in a word, and only any further parameters are passed on the stack. Consider:

```
compileString("""
        program parameterpassing;
          var a: boolean;
          var i, j: integer;
          procedure p(var u: boolean; v: boolean; w: integer;
                                var x, y: integer; z: integer);
            begin
              u := true; v := u and v; x := w + 3; w := 5; y := 7 - z
            end;
            begin p(a, true, 7, i, j, 9);
              if a then write(j)
            end
        """, 'P0.s')
```

Lab Question 1 (Annotating Generated Code, 8 points)**.** Add the missing annotations for the generated code below. This MIPS code has been generated by the P0 compiler after processing the above program:

```
        .data
j_:     .space 4
i_:     .space 4
a_:     .space 4
        .text
        .globl p
        .ent p
p:
                            # offset(u) = 20
                            # offset(v) = ?
                            # offset(w) = ?
                            # offset(x) = ?
                            # offset(y) = ?
                            # offset(z) = ?
        sw $fp, -28($sp)    # M[$sp - parametersize - 4] = $fp
        sw $ra, -32($sp)    # M[parametersize + 8 = 32
        sub $fp, $sp, 24    # $fp := $sp - parametersize
        sub $sp, $fp, 8     # $sp := $fp - 8
        lw $t0, 20($fp)     # $t0 := M[$fp + offset(u)]    [ = adr(u)]
        addi $t5, $0, 1     # $t5 := 1                     [ = true]
        sw $t5, 0($t0)      # [M[adr(u) = ] M[$t0] := $t5
        lw $t2, 20($fp)     # ?
        lw $t6, 0($t2)      # ?
        beq $t6, $0, L1     # ?
L2:
        lw $t8, 16($fp)     # ?
        beq $t8, $0, L3     # ?
```

```
L4:
        addi $t4, $0, 1    # ?
        b, L5              # ?
L3:
L1:
        addi $t4, $0, 0    # ?
L5:
        sw $t4, 16($fp)    # ?
        lw $t1, 8($fp)     # ?
        lw $t7, 12($fp)    # ?
        add $t7, $t7, 3    # ?
        sw $t7, 0($t1)     # ?
        addi $t3, $0, 5    # ?
        sw $t3, 12($fp)    # ?
        lw $t5, 4($fp)     # ?
        addi $t2, $0, 7    # ?
        lw $t6, 0($fp)     # ?
        sub $t2, $t2, $t6  # ?
        sw $t2, 0($t5)     # ?
        add $sp, $fp, 24   # ?
        lw $ra, -8($fp)    # ?
        lw $fp, -4($fp)    # ?
        jr $ra
        .text
        .globl main
        .ent main
main:
        la $t8, a_         # $t8 := adr(a)
        sw $t8, -4($sp)    # M[$sp - 4] := $t8
        addi $t4, $0, 1    # ?
        sw $t4, -8($sp)    # ?
        addi $t7, $0, 7    # ?
        sw $t7, -12($sp)   # ?
        la $t3, i_         # ?
        sw $t3, -16($sp)   # ?
        la $t2, j_         # ?
        sw $t2, -20($sp)   # ?
        addi $t6, $0, 9    # ?
        sw $t6, -24($sp)   # ?
        jal, p             # ?
        lw $t8, a_         # ?
        beq $t8, $0, L6    # ?
L7:
        lw $a0, j_         # ?
        li $v0, 1          # ?
        syscall
L6:
        li $v0, 10         # ?
        syscall
```

```
.end main
```