Assignment 3

Due: 9:00am, Tue Feb 6th, 2018

Note: Make reasonable assumptions where necessary and clearly state them. Feel free to discuss problems with classmates, but the only written material that you may consult while writing your solutions are the textbook and lecture slides/videos. Solutions should be uploaded as a single pdf file on Canvas. **Show your solution steps** so you receive partial credit for incorrect answers and we know you have understood the material. Don't just show us the final answer.

1. Read the assembly code below; add comments to explain what each line of code is doing; in one sentence, explain what this procedure is trying to accomplish. **(15 points)**

   ```
   new-proc:
     sll $a0, $a0, 24
     srl $a0, $a0, 24
     add $v0, $a0, $zero
     jr $ra
   ```

2. Read the assembly code below; add comments to explain what each of line of code is doing; provide a simple equation to express the return value v0 as a function of input arguments a0 and a1. **(35 points)**

   ```
   new-proc:
     blt $a1, $zero, loop2
   loop1:
     beq $a1, $zero, proc-end
     sll $a0, $a0, 1
     addi $a1, $a1, -1
     j loop1
   loop2:
     beq $a1, $zero, proc-end
     srl $a0, $a0, 1
     addi $a1, $a1, 1
     j loop2
   proc-end:
     add $v0, $a0, $zero
     jr $ra
   ```

3. For the (pseudo) assembly code below, replace X, Y, P, and Q with the smallest set of instructions to save/restore values on the stack and update the stack pointer. Assume that procA and procB were written independently by two different programmers who are following the MIPS guidelines for caller-saved and callee-saved registers. In other words, the two programmers agree on the input arguments and return value of procB, but they can't see the code written by the other person. Be sure to read the class notes first so you understand the MIPS guidelines for caller-saved and callee-saved registers. **(50 points)**

```
procA:
  $s0 = ...
  $s1 = ...
  $s2 = ...
  $t0 = ...
  $t1 = ...
  $t2 = ...
  X
  $a0 = ...
  $a1 = ...
  jal procB
  Y
  ... = $s1
  ... = $t0
  ... = $t1
  ... = $a0
  jr $ra

procB:
  P
  ... = $a0
  ... = $a1
  $s2 = ...
  $t0 = ...
  Q
  jr $ra
```