

Activity: Final Project Ideas; Subroutines

A. Why?

- The final project looms ever nearer.
- Even in assembler (especially in assembler?), subroutines are a good way to break up code into reusable pieces.

B. Outcomes

By the end of the class you should

- Have seen the ideas people have proposed for the final project
- Understand that saving registers without stacks precludes recursive subroutines.

C. Questions

1. Attached is a list of the final project ideas people proposed on Monday. In group, discuss which ones you like and why. If your group doesn't come to a consensus on which ideas it likes, that's okay. Write down which ideas your group members like. If you think of new ideas or have variations to suggest, write those down too.
2. In the textbook's sample programs in chapter 9, you see something like the following pattern. (The exact list of registers depends on the subroutine, though R7 and R0 are most popular.)

```

SUBR      ST      R0, SUBRSAVER0
          ST      R1, SUBRSAVER1
          ST      R7, SUBSAVER7

```

... useful work, possibly including a call to another subroutine:

```

          JSR     SUBR2

          LD      R7, SUBSAVER7
          LD      R1, SUBRSAVER1
          LD      R0, SUBRSAVER0
          RET     ; ie JMP R7
SUBRSAVR0 .BLKW 1
SUBRSAVR1 .BLKW 1
SUBRSAVR7 .BLKW 1

```

(2a) In general, why do we have the called routine save the registers it uses? (I.e., why not have the calling routine save the registers before making the call to SUBR?)

(2b) What happens if SUBR is recursive? How could we fix this problem?