Assignment 4, CS 223
Fall 2016
Parker Loomis
You are provided two files: A main and a function in separate source files. The function is a simple character swap, using the necessary pointers to accomplish 'pass by reference'. It also returns an integer just to prove it can.

This assignment will have you use gdb on the msp430 to answer questions about the structure and progress of the program. You will fill in answers to questions about the program. Please create a pdf file with those answers filled in. This can be done within a text editor that has a 'save as' option to pdf or one can use some pdf creator such as 'enscript'. It is this final pdf which I wish you to push back to me. Please remove the original source files, and any concomitant files (Words of Power !) which might have been created.

The purpose of this assignment is twofold:

1. Cure You of Insanity by 'encouraging' (forcing) you to use gdb

2. Look at the MSP430 architecture and how a working program is organized on the processor.

You are to answer the following questions, providing evidence via screen captures or the script program. You will be wise to have some documentation with you as you work through these questions. The header file <msp430.h> or the particular <msp430g2553.h>

I) Find the addresses of the following variables:

```
      1. P1DIR   &0x0021
      2. P1OUT   &0x0022
      3. str           0xc11c
      4. "A String"    0xc11e
      5. str_array     0xc084
      6. a       &0x0120
      7. b       0x0041
      8. tmp           4(r1),      r15   ;0x0004(r1)
      9. ptr1 (inside swap.c)      4(r1),     r15   ;0x0004(r1)
```

II) 1. Find the address of the statement `a = (char) 65`:
(Do this by setting a breakpoint and looking at the messages or literally look at R0)

0x0041

2. Find the address of the statement `tmp = *ptr1`

```
4(r1),      r15   ;0x0004(r1)
```

III) Parameters are SENT in registers R15, R14, R13, R12 (in order of argument list) and function values are returned in R15. Prove that the 'parameter passing protocol' above is honored in statements

1. i = swap(&a, &b);

```
0x0000c060 <+34>:     mov.b #65,  4(r1) ;#0x0041, 0x0004(r1)
   0x0000c066 <+40>:  mov.b #126, 5(r1) ;#0x007e, 0x0005(r1)
   0x0000c06c <+46>:  mov   r1,   r14
   0x0000c06e <+48>:  add   #5,   r14   ;#0x0005
   0x0000c072 <+52>:  mov   r1,   r15
   0x0000c074 <+54>:  add   #4,   r15   ;r2 As==10

2. i = swap(&str_array[2], &str_array[5]);
 0x0000c07e <+64>:    mov   #-16100,2(r1)    ;#0xc11c, 0x0002(r1)
   0x0000c084 <+70>:  mov.b #65,  6(r1) ;#0x0041, 0x0006(r1)
   0x0000c08a <+76>:  mov.b #32,  7(r1) ;#0x0020, 0x0007(r1)
   0x0000c090 <+82>:  mov.b #83,  8(r1) ;#0x0053, 0x0008(r1)
   0x0000c096 <+88>:  mov.b #116, 9(r1) ;#0x0074, 0x0009(r1)
   0x0000c09c <+94>:  mov.b #114, 10(r1)     ;#0x0072, 0x000a(r1)
---Type <return> to continue, or q <return> to quit---
   0x0000c0a2 <+100>: mov.b #105, 11(r1)     ;#0x0069, 0x000b(r1)
   0x0000c0a8 <+106>: mov.b #110, 12(r1)     ;#0x006e, 0x000c(r1)
   0x0000c0ae <+112>: mov.b #103, 13(r1)     ;#0x0067, 0x000d(r1)
   0x0000c0b4 <+118>: mov.b #0,   14(r1)     ;r3 As==00, 0x000e(r1)
=> 0x0000c0b8 <+122>: mov   r1,   r15
   0x0000c0ba <+124>: add   #6,   r15   ;#0x0006
   0x0000c0be <+128>: mov   r15,  r14
   0x0000c0c0 <+130>: add   #5,   r14   ;#0x0005
   0x0000c0c4 <+134>: mov   r1,   r15
   0x0000c0c6 <+136>: add   #6,   r15   ;#0x0006
```