Due: Programs: Monday July 4th at 11:59pm. Written Tuesday July 5th at 4:00pm.
Filenames: authors.csv, ch3-8.csp, ch4-12.csp, ch5-14.csp, ch5-17a.csp and ch5-17b.csp, ch6-11.csp, ch6-12.csp

Format of authors.csv: author1_email,author1_last_name,author1_first_name
author2_email,author2_last_name,author2_first_name

You must submit the written answers in the homework box in 2131 Kemper. Each student must submit his/her own written answers. You must use the handin program to submit the seven programs to the p2 directory of cs50. All of the programs of each team should be submitted from exactly one account. You must name your programs the names shown here. Make sure that your programs may be rerun without reloading the .OBJ file; this means that you must initialize counter variables in the program and NOT use assembler directives. You will find executables created from my .c files in ~/ssdavis/50/p2.

Written (40 points, 2 points for each part)

Chapter #3 Exercises: 3 a, c; 4 b, d;
Chapter #4 Exercises: 4; 6 a, c; 7 a, c;
Chapter #5 Exercises: 8, 9
Chapter #6 Exercises: 3; 4; 7 b, d; 8 b, d; 9 b, d; 10.

Programs (60 points, 10 points each except ch5-17a.csp and ch5-17b.csp)

All functions, other than main(), must be register neutral. I have provided C programs for the Chapter 3 and 4 programs in ~/ssdavis/50/p2. To ease the work of the reader for the programs from Chapters 5 and 6, we will supply the initial array(s) in the area $A00 to $B06. Do NOT write to this area in any of your code! I have written a C program that creates a separate file that stores values in $A00. The array we test with may have up to 250 elements. You will find CreateArray.out, and three such array files in ~/ssdavis/50/p2.

For the two Chapter 5 assigned programs, 14 and 17. We will supply the initial arrays, along with their dimensions when testing both questions. At the top of both programs you should have the following:

.EQU ROWS, $A00 ; # of rows in the array. Note that the index would be 0 to ROWS - 1.
.EQU COLS, $A01 ; # of columns in the array (not used for #14).
.EQU ARR, $A02 ; Beginning of the initial array

I suggest you copy the following .equ’s to the top of all of the files. You will not be using them all in any program, but it won’t hurt to have them available.

.EQU PUT_NUM, $E00
.EQU GET_NUM, $E01
.EQU PUT_STR, $E05
.EQU PUT_NL, $E06
.EQU PUT_NUM2, $E07
.EQU PUT_CHR, $E08

Note: The string following a .CHAR directive should be in single quotes, and not double quotes as indicated on page 96 of the book. For example, for ch4-12.csp you would use:
PROMPT1: .CHAR 'Please enter action', PROMPT1_LEN

1. Chapter 3 #8, Filename: ch3-8.csp Your program should continue to input numbers, construct and display the 24-bit even parity number until the user enters zero. Make sure that your program can be rerun without reload. You will find that you will need to store $800000 using the .WORD directive. Use JSR GET_NUM to read the value from the user, and use JSR PUT_NUM to write the result to the screen.
2. Chapter 4 #12, Filename: ch4-12.csp The C version for this one emulates the 24 bit CUSP OV system and raises unnecessary issues for our 32-bit systems. So don’t be misled by the manipulating of total at the end of the code. Your CUSP code will not need that.

3. Chapter 5 #14 Filename: ch5-14.csp You will only use the first “row” of the array ARR as the source for your own partial summing array, S. The number of entries is given by ROWS. The COLS value is completely ignored for this program. You need only set aside 50 words for array S. After you have computed S, print it all out on one line, with each sum separated by one space.

4. Chapter 5 #17 Filenames: ch5-17a.csp and ch5-17b.csp (5 points each) Do not bother creating another array. Just print out the values of ARR[i, j] * 100 / MAX. As in Figure 5.13, print out each row on a separate line with one space between each number.

   Given the following:
   .EQU @, $A00
   .WORD 4 ; ROWS
   .WORD 2 ; COLS
   .WORD 1 ; ARR
   .WORD 6
   .WORD 3
   .WORD 4
   .WORD 12
   .WORD 25
   .WORD 14
   .WORD 10

   The output of #14 would be:
   1 7 10 14
   
   The output of #17 would be:
   4 24
   12 16
   48 100
   56 40

5. Chapter 6 #11. Filename: ch6-11.csp Your LOAD procedure should prompt and receive from the user the number of elements in the array. Then just copy from the array starting at $A00 into your own array. You need not worry about testing for MAX_N.

6. Chapter 6 #12. Filename: ch6-12.csp You will find cnvtnode.csp in ~ssdavis/50/p2, and may copy it into your ch6-12.csp. There are no changes to the specifications. Just make sure you do not use the area $A00 to $B06 for your own storage.