6. PROGRAMMING ASSIGNMENT 6:  UPDATED 7/20/17
Read:       Chapter 7 (Read: pages 137-152, Skip: pages 259-265)
Programming: Name your program p6.c
DUE:        Friday July 28, 2017 6am

This program will use one-dimensional arrays and functions to calculate and display midterm’s average, highest, and descending sorted order using array and pointer notation.

a) To increase readability, use symbolic constants e.g. “
   #define NUM_SCORE 5
   #define MAX_SCORE 103 ”

b) main() defines an integer array initialized to:  72 78 82 88 99
   After calculating average and sorted order then input is requested with getScores(). A flag specifying first run may help for calling getScores().

c) You will write 5 functions besides main(). main() will call (invoke) all functions except swap(). The function declarations are:
   void   getScores (int a[]);
   void   bubbleSort(int a[]);
   void   prtScores (int *p);
   double average   (int a[]);
   void   swap      (int *p, int *q);

d) getScores() will read input into the 1-dimensional array. Display an error message if the integer input value of 0 through 103 for the midterm score is not in this range. Prompt for another FIVE integers to be input in the correct range. A loop will repeat the prompt until ALL five valid integers are entered. Do not calculate the statistics for an invalid input number. Call other functions only upon valid input. See sample output for example. Hint: An integer "flag" (e.g. error_input = 1;) may help in the looping condition.

   getscores() will read input into temporary variables, i.e. tmp. Test if tmp is within the valid input value of 0 through 103. If it is not in this range, then prompt again for ALL 5 input values to be typed in again. Repeat input and test until a valid value is entered (do-while works fine).

   If tmp is a valid value, the assign into the corresponding elements of the 1-dimensional array. Call other functions only upon valid input. See sample output for example.

e) prtScores() using pointer notation will display the values of the 1-dimensional array.
f) average() using array notation will calculate the average of all the scores.
g) bubbleSort() will use a “bubble sort” algorithm by bubbling up the largest value into first element. prtScores() will print sorted array.
h) swap() is same as in PA#5.

i) Your program must loop so that the user can continue to generate and view the output until the user responds with 'n' or 'N' to the offer "Want to calculate midterm scores?". NOTE: Your program will end ONLY with the input of 'n' or 'N'.

HINTS:  Solve this problem in small steps. Here’s a suggestion.
1) In main(), write the code for step a) and b) above. Test.
2) Write the code for step e) above, prtScores() using array notation. Modify later into pointer notation.
HINTS: (continued from previous page)

3) Write the code for step f) above, `average()` . Refer to your Arrays handout.

4) Write the code for step g) above ( `bubbleSort()` ).

5) Type in code for swap() from PA#5. In main(), display the highest score.

6) Write the code for step d) above ( `getScores()` ). Test, then code for error input later.

7) Write the code for step i) above so your entire program will loop again.

8) Convert `prtScores()` into pointer notation.

9) Complete error checking. DONE!

**PA#6 SAMPLE OUTPUT:**

```
Array:          72 78 82 88 99
Average:        83.80
Sorted:         99 88 82 78 72
Highest:        99

Want to calculate midterm scores? a

Enter 5 midterm scores (0-103): 68 88 77 98 58
Average:        77.80
Sorted:         98 88 78 68 58
Highest:        98

Want to calculate midterm scores? b

Enter 5 midterm scores (0-103): -3 93 333 83 73
ERROR! Enter valid scores in the range (0-103)
ERROR! Enter valid scores in the range (0-103)

Enter 5 midterm scores (0-103): 22 92 82 103 33
Average:        66.20
Sorted:         103 92 82 33 22
Highest:        103

Want to calculate midterm scores? N
```

**PA#6 SAMPLE INPUT** (typed in bold above)

Submit the final version of your program as “p6.c”.

Verify you saved your work in the Documents - cs5u HOME directory.

```c
void  bubbleSort( int a[] )
{
  int  i, j;

  for(i = 0 ; i < NUM_SCORE-1 ; ++i)
    for(j = NUM_SCORE-1 ; i < j ; --j)
      if( a[j-1] > a[j] )
        swap(&a[j-1], &a[j]);
}
```