Exercise 0 *(6 points – 1 point per question – No program required)*

1. A
2. C
3. D
4. C
5. C
6. D

References and Explanations:
In addition to the course book references cited below, these topics are also covered in the live lectures (in-class students) and the recorded lectures (online students).

1. Note 6.1; Determining the correct answer is simply a matter of using the "Right-Left" rule.

2. Note 1.1; Neither C nor C++ have array boundary checking. As a result, if an array element is accessed that is not actually in that array, such as by using a negative index value or an index value greater than the total number of elements minus 1, a region of memory will be accessed that the code has no right to access. The result can be anything from crashing the program entirely if you are extremely lucky (which usually doesn't happen) to getting an apparently good value (that is nonetheless a garbage value) if you are extremely unlucky (which usually does happen). By definition a "garbage" value is any value that is unpredictable or obtained via an illegal operation (including 0).

3. Note 5.11; The most important thing that can be said about any code concerns issues that can cause erratic operation and/or erroneous results. In this code pointer variable *p* is declared but never initialized, and as a result will contain a garbage address. Thus, when the **p = 25.6**L*** gets executed *25.6**L*** will be stored into the garbage address represented by *p*. Depending upon what that address actually represents, this can cause program crashes, program data corruption, or no bad effects at all.

4. Notes 6.5, 6.6, 6.7, 6.9, 6.10; When used as a unary operator the address operator, **&**, produces the address of (pointer to) its operand. In the expression *fcnA(&x, &y)* the addresses of variables *x* and *y* are being passed as arguments. Since addresses and pointers are one in the same thing, it is also correct and more common to say that pointers to variables *x* and *y* are being passed.

5. Note 6.10; Either of a pointer to a non-**const** or a pointer to a const may be passed to a function whose corresponding parameter is a pointer to a **const**. However, only a pointer to a non-**const** may be passed to a function whose corresponding parameter is a pointer to a non-**const**. The left argument of *fcnA(&x, &y)* violates the last part of this requirement.

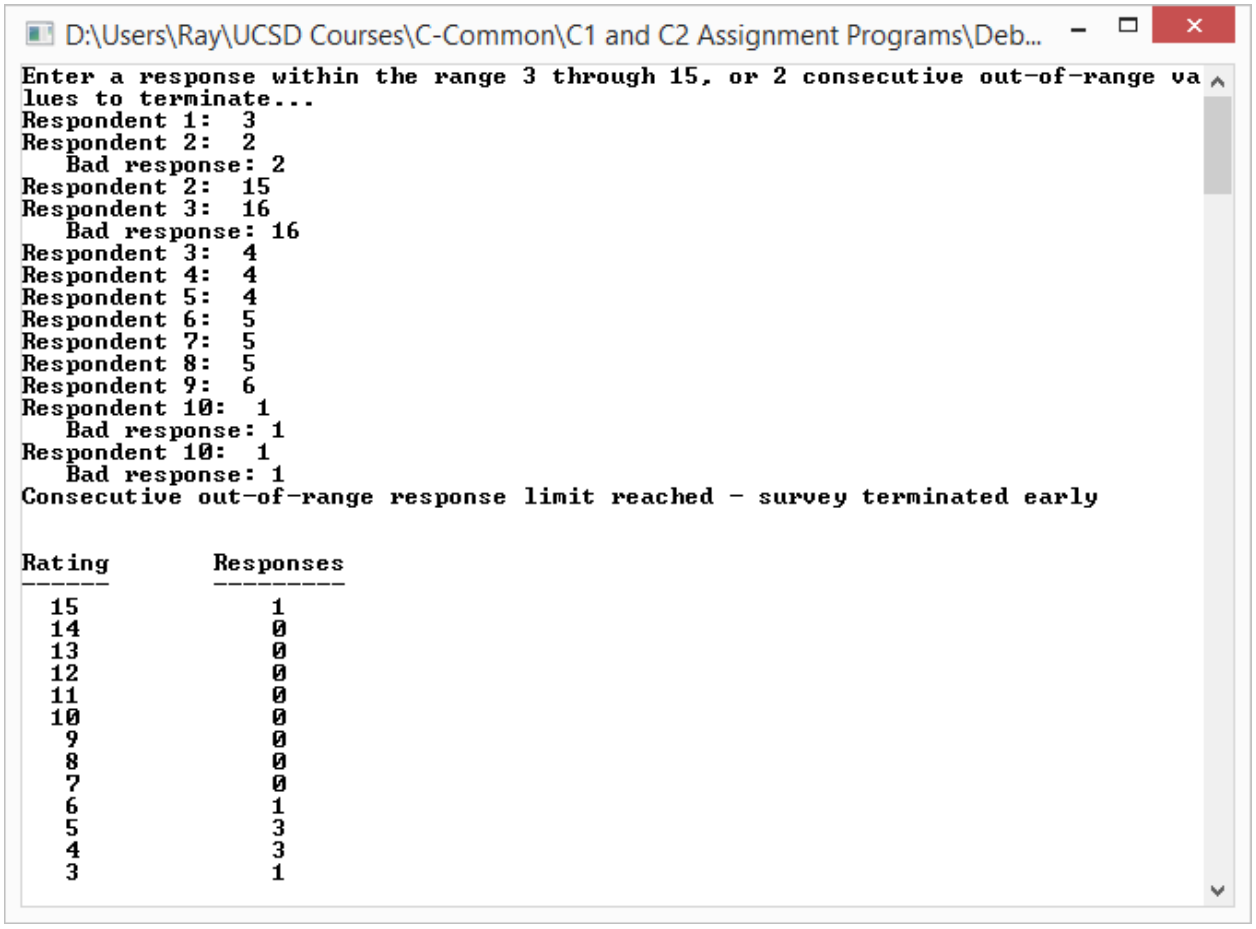
6. Notes 5.11, 6.12; An automatic variable is any variable declared inside a block without either of the keywords **static** or **extern**. An automatic variable becomes invalid when the block in which it is declared is exited. The automatic variable *x* in this code sample is declared inside the block that forms the body of the function and it becomes invalid when the function returns. Thus, its address also becomes invalid at that time and only represents the address where that variable used to be. It is illegal to dereference any invalid address.

Exercise 1 (6 points – C Program)

```c
/*
 * ...the usual title block Student/Course/Assignment/Compiler information goes here...
 *
 * This file contains function main, which prompts the user for values for
 * a survey and displays the results.
 */

#include <stdio.h>
#include <stdlib.h>

#define MAX_RESPONDENTS     17   /* maximum respondents */
#define MIN_RESPONSE_VALUE (-27) /* minimum response value */
#define MAX_RESPONSE_VALUE   9   /* maximum response value */
#define OUT_OF_RANGE_LIMIT   1   /* # of bad responses until terminate */

/* Number of possible response values */
#define RESPONSE_VALUES (MAX_RESPONSE_VALUE - MIN_RESPONSE_VALUE + 1)

/*
 * Prompt user up to MAX_RESPONDENTS times to input an integral value in
 * the range [MIN_RESPONSE_VALUE, MAX_RESPONSE_VALUE], keeping count of
 * how many times each response value has been input.  MIN_RESPONSE_VALUE
 * and MAX_RESPONSE_VALUE may be negative.  Terminate prompting and display
 * the number of each possible response value that has occurred when either
 * MAX_RESPONDENTS "in range" legal responses have occurred or when
 * OUT_OF_RANGE_LIMIT "out-of-range" responses in a row have occurred.
 */
int main(void)
{
   int ratingCounters[RESPONSE_VALUES] = {0};
   int respCounter, consecutiveRangeErrors = 0;

   printf("Enter a response within the range %d through %d, "
      "or %d consecutive out-of-range values to terminate...\n",
      MIN_RESPONSE_VALUE, MAX_RESPONSE_VALUE, OUT_OF_RANGE_LIMIT);
   for (respCounter = 0; respCounter < MAX_RESPONDENTS;) /* response loop */
   {
      int response;
      printf("Respondent %d:  ", respCounter + 1);       /* prompt user */
      scanf("%d", &response);                            /* get response */
      if (response < MIN_RESPONSE_VALUE || response > MAX_RESPONSE_VALUE)
      {
         fprintf(stderr, "   Bad response: %d\n", response);   /* error msg */
         if (++consecutiveRangeErrors >= OUT_OF_RANGE_LIMIT)
            break;
      }
      else                                               /* legal response */
      {
         ++ratingCounters[response - MIN_RESPONSE_VALUE];/* inc rating count */
         ++respCounter;                                  /* next respondent */
         consecutiveRangeErrors = 0;                     /* no bad responses */
      }
   }
   if (consecutiveRangeErrors < OUT_OF_RANGE_LIMIT)
      printf("All respondents polled: survey terminated\n");
```

```c
    else
       fprintf(stderr, "Consecutive out-of-range response limit reached - "
          "survey terminated early\n");


    /* For each rating, display the number of respondents... */
    printf("\n\n"
       "Rating       Responses\n"                       /* print resp... */
       "------       ---------\n");                     /* ...table header */
    for (respCounter = RESPONSE_VALUES - 1; respCounter >= 0; --respCounter)
       printf("%4d%14d\n",
          respCounter + MIN_RESPONSE_VALUE, ratingCounters[respCounter]);

    return EXIT_SUCCESS;
}
```

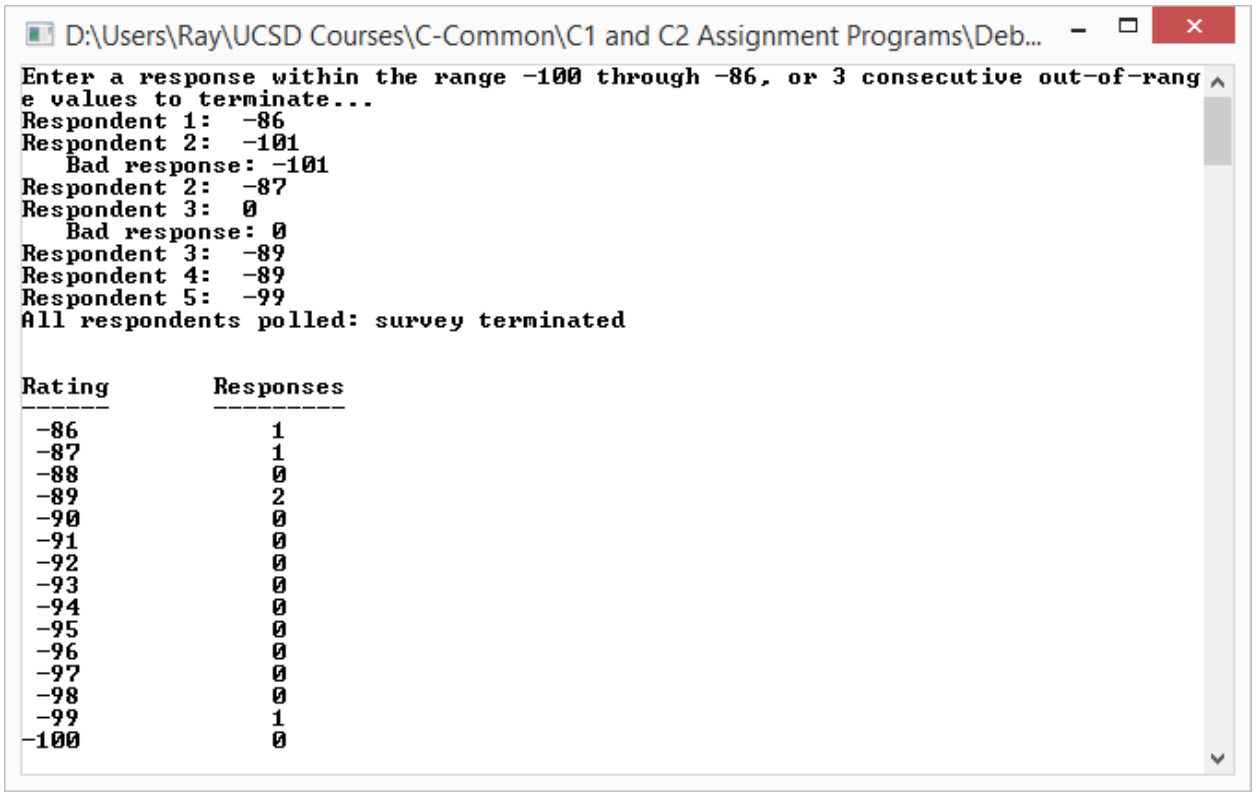
C1A5E1 Screen Shots


D:\Users\Ray\UCSD Courses\C-Common\C1 and C2 Assignment Programs\Deb...

```
Enter a response within the range 3 through 15, or 2 consecutive out-of-range va
lues to terminate...
Respondent 1:   3
Respondent 2:   2
   Bad response: 2
Respondent 2:   15
Respondent 3:   16
   Bad response: 16
Respondent 3:   4
Respondent 4:   4
Respondent 5:   4
Respondent 6:   5
Respondent 7:   5
Respondent 8:   5
Respondent 9:   6
Respondent 10:   1
   Bad response: 1
Respondent 10:   1
   Bad response: 1
Consecutive out-of-range response limit reached - survey terminated early


Rating       Responses
------       ---------
  15             1
  14             0
  13             0
  12             0
  11             0
  10             0
   9             0
   8             0
   7             0
   6             1
   5             3
   4             3
   3             1
```

C1A5E1 Screen Shots continue on the next page…

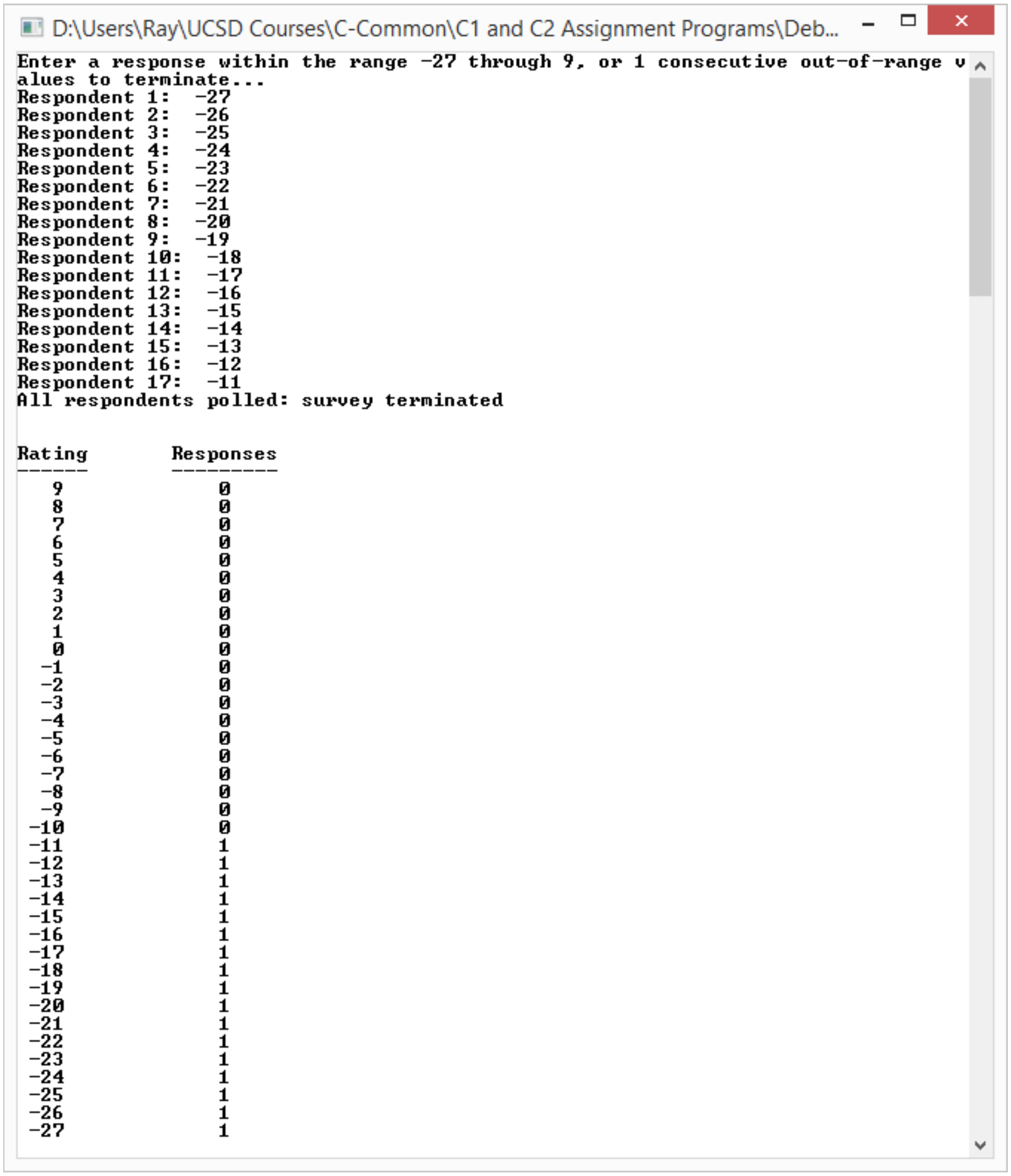
C1A5E1 Screen Shots, continued


D:\Users\Ray\UCSD Courses\C-Common\C1 and C2 Assignment Programs\Deb...

```
Enter a response within the range -100 through -86, or 3 consecutive out-of-rang
e values to terminate...
Respondent 1:  -86
Respondent 2:  -101
   Bad response: -101
Respondent 2:  -87
Respondent 3:  0
   Bad response: 0
Respondent 3:  -89
Respondent 4:  -89
Respondent 5:  -99
All respondents polled: survey terminated


Rating       Responses
------       ---------
 -86             1
 -87             1
 -88             0
 -89             2
 -90             0
 -91             0
 -92             0
 -93             0
 -94             0
 -95             0
 -96             0
 -97             0
 -98             0
 -99             1
-100             0
```

C1A5E1 Screen Shots continue on the next page…

C1A5E1 Screen Shots, continued


D:\Users\Ray\UCSD Courses\C-Common\C1 and C2 Assignment Programs\Deb...

```
Enter a response within the range -27 through 9, or 1 consecutive out-of-range v
alues to terminate...
Respondent 1:  -27
Respondent 2:  -26
Respondent 3:  -25
Respondent 4:  -24
Respondent 5:  -23
Respondent 6:  -22
Respondent 7:  -21
Respondent 8:  -20
Respondent 9:  -19
Respondent 10:  -18
Respondent 11:  -17
Respondent 12:  -16
Respondent 13:  -15
Respondent 14:  -14
Respondent 15:  -13
Respondent 16:  -12
Respondent 17:  -11
All respondents polled: survey terminated


Rating       Responses
------       ---------
   9             0
   8             0
   7             0
   6             0
   5             0
   4             0
   3             0
   2             0
   1             0
   0             0
  -1             0
  -2             0
  -3             0
  -4             0
  -5             0
  -6             0
  -7             0
  -8             0
  -9             0
 -10             0
 -11             1
 -12             1
 -13             1
 -14             1
 -15             1
 -16             1
 -17             1
 -18             1
 -19             1
 -20             1
 -21             1
 -22             1
 -23             1
 -24             1
 -25             1
 -26             1
 -27             1
```

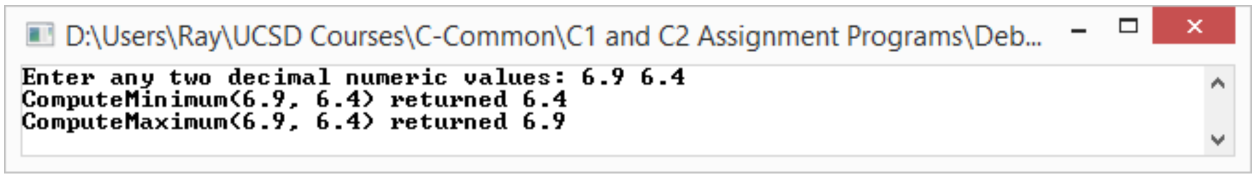
Exercise 2 *(4 points – C++ Program)*

```
    *************************************** FILE C1A5E2_main.cpp ***************************************
//
// ...the usual title block Student/Course/Assignment/Compiler information goes here...
//
// This file contains function main, which prompts the user for two values,
// calls functions that determine their maximum and minimum using references,
// then displays the results.
//

#include <iostream>
using std::cin;
using std::cout;

// Prototypes for custom functions called by main.
double &ComputeMaximum(const double &val1, const double &val2);
double &ComputeMinimum(const double &val1, const double &val2);

//
// Test functions ComputeMinimum and ComputeMaximum by calling them with
// values obtained from user input and display the results.  These functions
// must return references to the minimum and maximum of the two values whose
// references are passed to them as arguments.
//
int main()
{
   double value1, value2;
   // Get two user input values and determine which is lesser/greater.
   cout << "Enter any two decimal numeric values: ";
   cin >> value1 >> value2;

   cout << "ComputeMinimum(" << value1 << ", " << value2 << ") returned " <<
      ComputeMinimum(value1, value2) << "\n";
   cout << "ComputeMaximum(" << value1 << ", " << value2 << ") returned " <<
      ComputeMaximum(value1, value2) << "\n";

   return 0;
}
```

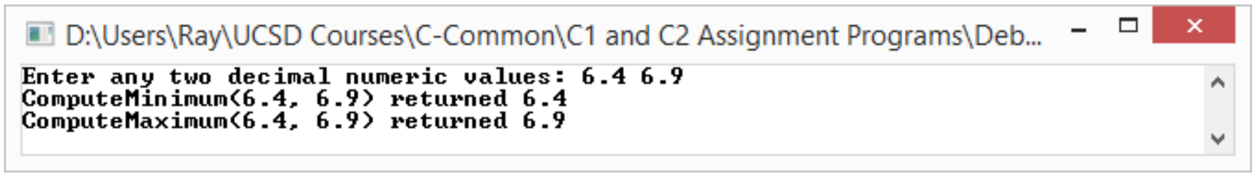
-- EXERCISE CONTINUES ON NEXT PAGE --

```

   ********************************** FILE C1A5E2_ComputeMinimum.cpp **********************************
//
// ...the usual title block Student/Course/Assignment/Compiler information goes here...
//
// This file contains function ComputeMinimum, which returns a reference
// to the minimum of the two values referenced by its parameters.
//

//
// Determine the minimum of the values referenced by its two parameters and
// return a reference to it.  No special test is needed for equal values.
//
double &ComputeMinimum(const double &val1, const double &val2)
{
   // Compare <val1> and <val2> and return the minimum.
   return (double &)(val1 < val2 ? val1 : val2);
}


   ********************************* FILE C1A5E2_ComputeMaximum.cpp *********************************
//
// ...the usual title block Student/Course/Assignment/Compiler information goes here...
//
// This file contains function ComputeMaximum, which returns a reference
// to the maximum of the two values referenced by its parameters.
//

//
// Determine the maximum of the values referenced by its two parameters and
// return a reference to it.  No special test is needed for equal values.
//
double &ComputeMaximum(const double &val1, const double &val2)
{
   // Compare <val1> and <val2> and return the maximum.
   return (double &)(val1 > val2 ? val1 : val2);
}
```

C1A5E2 Screen Shots


D:\Users\Ray\UCSD Courses\C-Common\C1 and C2 Assignment Programs\Deb...

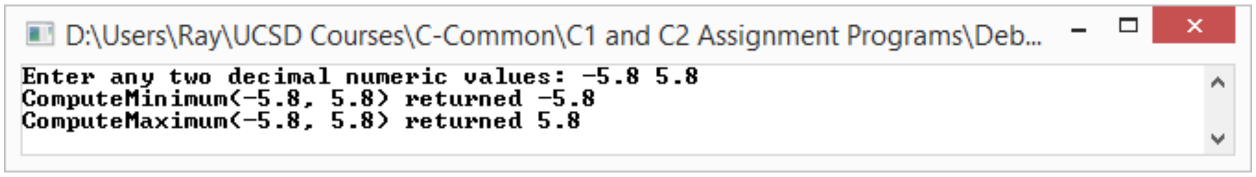
```
Enter any two decimal numeric values: 6.9 6.4
ComputeMinimum(6.9, 6.4) returned 6.4
ComputeMaximum(6.9, 6.4) returned 6.9
```


D:\Users\Ray\UCSD Courses\C-Common\C1 and C2 Assignment Programs\Deb...

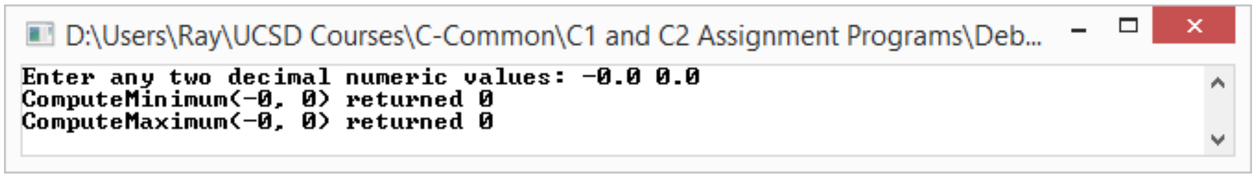
```
Enter any two decimal numeric values: 6.4 6.9
ComputeMinimum(6.4, 6.9) returned 6.4
ComputeMaximum(6.4, 6.9) returned 6.9
```

C1A5E2 Screen Shots continue on the next page...

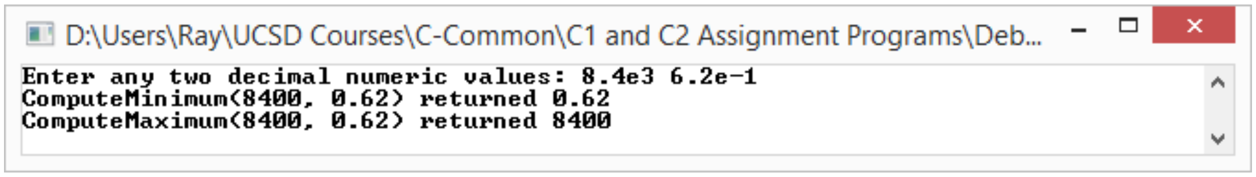
C1A5E2 Screen Shots, continued


D:\Users\Ray\UCSD Courses\C-Common\C1 and C2 Assignment Programs\Deb...

```
Enter any two decimal numeric values: -5.8 5.8
ComputeMinimum(-5.8, 5.8) returned -5.8
ComputeMaximum(-5.8, 5.8) returned 5.8
```


D:\Users\Ray\UCSD Courses\C-Common\C1 and C2 Assignment Programs\Deb...

```
Enter any two decimal numeric values: -0.0 0.0
ComputeMinimum(-0, 0) returned 0
ComputeMaximum(-0, 0) returned 0
```


D:\Users\Ray\UCSD Courses\C-Common\C1 and C2 Assignment Programs\Deb...

```
Enter any two decimal numeric values: 8.4e3 6.2e-1
ComputeMinimum(8400, 0.62) returned 0.62
ComputeMaximum(8400, 0.62) returned 8400
```

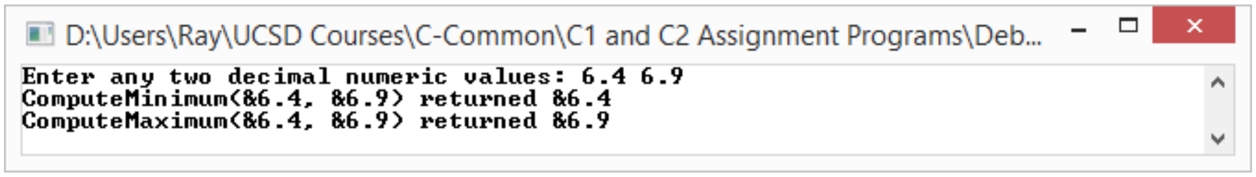
Exercise 3 *(4 points – C++ Program)*

```
    ****************************************  FILE C1A5E3_main.cpp  ****************************************
//
// ...the usual title block Student/Course/Assignment/Compiler information goes here...
//
// This file contains function main, which prompts the user for two values,
// calls functions that determine their maximum and minimum using pointers,
// then displays the results.
//

#include <iostream>
using std::cin;
using std::cout;

// Prototypes for custom functions called by main.
double *ComputeMaximum(const double *val1, const double *val2);
double *ComputeMinimum(const double *val1, const double *val2);

//
// Test functions ComputeMinimum and ComputeMaximum by calling them with
// values obtained from user input and display the results.  These functions
// must return pointers to the minimum and maximum of the two values whose
// pointers are passed to them as arguments.
//
int main()
{
   double value1, value2;
   // Get two user input values and determine which is lesser/greater.
   cout << "Enter any two decimal numeric values: ";
   cin >> value1 >> value2;

   cout << "ComputeMinimum(&" << value1 << ", &" << value2 << ") returned &" <<
      *ComputeMinimum(&value1, &value2) << "\n";
   cout << "ComputeMaximum(&" << value1 << ", &" << value2 << ") returned &" <<
      *ComputeMaximum(&value1, &value2) << "\n";

   return 0;
}
```

--- EXERCISE CONTINUES ON NEXT PAGE ---

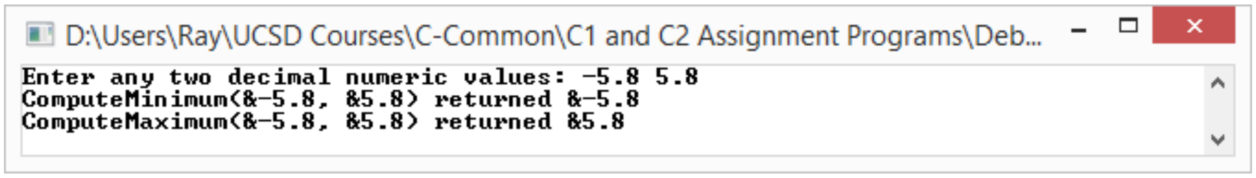
```
   ********************************** FILE C1A5E3_ComputeMinimum.cpp **********************************
//
// ...the usual title block Student/Course/Assignment/Compiler information goes here...
//
// This file contains function ComputeMinimum, which returns a pointer
// to the minimum of the two values pointed to by its parameters.
//

//
// Determine the minimum of the values pointed to by its two parameters and
// return a pointer to it. No special test is done for the values being equal.
//
double *ComputeMinimum(const double *val1, const double *val2)
{
   // Compare <*val1> and <*val2> and return the minimum.
   return (double *)(*val1 < *val2 ? val1 : val2);
}


   ********************************* FILE C1A5E3_ComputeMaximum.cpp *********************************
//
// ...the usual title block Student/Course/Assignment/Compiler information goes here...
//
// This file contains function ComputeMaximum, which returns a pointer
// to the maximum of the two values pointed to by its parameters.
//

//
// Determine the maximum of the values pointed to by its two parameters and
// return a pointer to it. No special test is done for the values being equal.
//
double *ComputeMaximum(const double *val1, const double *val2)
{
   // Compare <*val1> and <*val2> and return the maximum.
   return (double *)(*val1 > *val2 ? val1 : val2);
}
```

C1A5E3 Screen Shots



D:\Users\Ray\UCSD Courses\C-Common\C1 and C2 Assignment Programs\Deb...

```
Enter any two decimal numeric values: 6.4 6.9
ComputeMinimum(&6.4, &6.9) returned &6.4
ComputeMaximum(&6.4, &6.9) returned &6.9
```

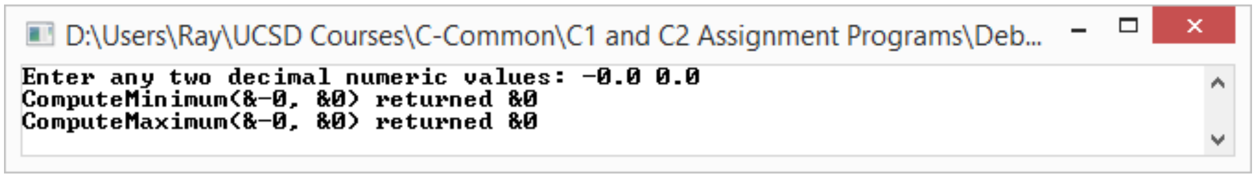


D:\Users\Ray\UCSD Courses\C-Common\C1 and C2 Assignment Programs\Deb...

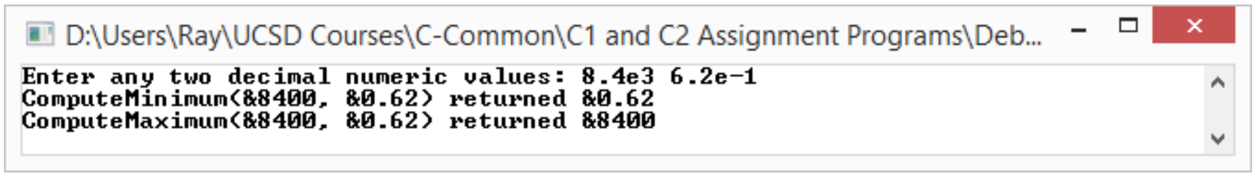
```
Enter any two decimal numeric values: -5.8 5.8
ComputeMinimum(&-5.8, &5.8) returned &-5.8
ComputeMaximum(&-5.8, &5.8) returned &5.8
```

C1A5E3 Screen Shots continue on the next page…

C1A5E3 Screen Shots, continued


D:\Users\Ray\UCSD Courses\C-Common\C1 and C2 Assignment Programs\Deb...

```
Enter any two decimal numeric values: -0.0 0.0
ComputeMinimum(&-0, &0) returned &0
ComputeMaximum(&-0, &0) returned &0
```


D:\Users\Ray\UCSD Courses\C-Common\C1 and C2 Assignment Programs\Deb...

```
Enter any two decimal numeric values: 8.4e3 6.2e-1
ComputeMinimum(&8400, &0.62) returned &0.62
ComputeMaximum(&8400, &0.62) returned &8400
```