Problem: You are developing a robot that can be pre-programmed to move around a room. At this point in time you are only working on moving the robot in one dimension (along a single axis). The starting location of the robot will be considered the furthest point that the robot can move in the left direction and the ending location the furthest that can be moved to the right. After the user enters the starting and ending points they will then enter how they want the robot to move with each positive value entered representing moving that many points to the right and a negative number is moving to the left. The end of the input for the moving of the robot will be represented by an entry of zero. Display the final location of the robot in the room.

Example Execution #1:

Enter starting address: 10
Enter ending address: 20
Enter values to move now: 5 4 -3 0

Final location: 16

Example Execution #2 (the robot cannot progress beyond point 15 and will remain there until instructed to move left):

Enter starting address: 0
Enter ending address: 15
Enter values to move now: 5 5 5 5 -5 0

Final location: 10

Example Execution #3 (similar to example #2, robot hits limit on left):

Enter starting address: 0
Enter ending address: 15
Enter values to move now: 5 5 -5 -5 -5 5 0

Final location: 5

Example Execution #4:

Enter starting address: -4
Enter ending address: 10
Enter values to move now: 1 1 1 1 1 1 1 1 1 0

Final location: 6

Example Execution #5 (input validation requirements demonstrated):

Enter starting address: 10
Enter ending address: 5

Error! Ending address must be >= 10

Enter ending address: 6

Error! Ending address must be >= 10

Enter ending address: 11
Enter values to move now: 1 1 1 1 1 1 1 1 1 1 -1 0

Final location: 10
Additional Requirements:

1. For this assignment you will be required to implement the user-defined functions (from chapter 4).
   - Revisit course standards as it relates what makes for good use of user-defined functions, what is acceptable to retain in the **main** function, and when passing parameters by address is appropriate.
   - Failing to follow course standards as they relate to good user-defined function use may result in a **zero for this assignment.**

2. Only material from the first SIX chapters of the C programming text may be used in this assignment.
   - Selection is acceptable (chapter 5).
   - Repetition is acceptable (chapter 6).
   - Arrays/strings are **not acceptable** (chapter 8).

3. Your program must accept input and produce output in the same manner (**exactly**) demonstrated in the example executions. Each example represents a single execution of your program.
   - See example #5 for the input validation requirements. Review in the course packet the expectations for validating user input.
   - All data will be of the integer type.

Academic Integrity Reminder:

- Please review the policies of the course as they relate to academic integrity. The assignment you submit should be your own original work. You should be consulting only course staff regarding your specific algorithm for assistance. Collaboration is not permitted on individual homework assignments.

Course Programming and Documentation Standards Reminders:

- Place a **single space** between all operators and operands. Comment **all** variables to the right of each declaration.
- Declare only one variable per line. Select **meaningful identifiers** (names) for all variables in your program.
- **Indent** all code found within the **main** function and all user-defined functions **exactly** two spaces.
- All code found within another construct that contains a body (if/else if/else, loops) must be **indented exactly** two additional spaces.
- Make good use of symbolic/defined constants. Minimize your use of literal constant values.
- Use the course function header (**head_fx vi** shortcut **hfx** while in command mode) for every user-defined function in your program.
  - List and comment **all parameters** to a function, one per line, in the course function header.
  - **All function declarations** will appear in the global declaration section of your program.
  - **The user-defined function definitions will appear in your program after the main function.**

When you submit... only the final attempt of a submission is kept for grading. All other submissions are over-written and cannot be recovered. You may make multiple submissions but only the last attempt is retained and graded.

- Verify in the confirmation e-mail sent to you by the course that you have submitted the correct file, to the correct assignment (**hw05**), and to the correct lab section.
- Leave time prior to the due date to seek assistance should you experience difficulties completing or submitting this assignment.
- All attempts to submit via a method other than through the **guru** server as set up during the first week of the semester will be denied consideration.

Assignment deadlines... are firm and the electronic submission will disable promptly as advertised. We can only grade what you submit as expected prior to the assignment deadline.

All course programming and documentation standards are in effect for this and each assignment this semester. Please review this document!