Problem 1 (20%)  
Write a handy board program that takes inputs for the A, B, C, D and then outputs the result, E to the screen. The result should be computed using the logic circuit described below.

The program should start immediately when the handy board is turned on. Pressing the start button will indicate a 1 has been entered. Pressing stop will indicate a 0. After A has been input, turn on the top LED green if it was a 1 and red if it was a 0. The next time start or stop is pressed, B will be input. Light up the second LED from the top accordingly. After all four inputs have been specified, calculate the result and print it to the handy board’s screen.

Hint: Even though this is not a design problem, consider a highly modular approach to save time.

Turn In: Printout of all source code and copy your program as username_PS6_P1.c to \CDIO-Prime\16.070HW\ps6\p1\. Indicate how much time you spent on this problem.
Problem 2 (40%)

For this task, you're going to implement a Substitution Cypher to decode a message. The tasks in the problem will guide you in modifying a short program, using arrays, and give you a preview of character input and manipulation.

The Cypher:
A simple substitution cypher encrypts a message by mapping each unique letter in the plaintext to a different letter in the alphabet. As you encrypt the message, you write down a key so you can decrypt it later. The first letter in the key is the cypher for 'A'; the second letter is the cypher for 'B' and so on.

For example, if my key for the first nine letters of the alphabet was \[ W Z E N H J X Q B \ldots \], the message “H I  D A D!” would be encrypted “Q B  N W N!”

You can use an array of characters in C to hold the key. To encrypt a message, map each letter to a number between 0 and 25, and find the letter at that index in the array. (A is the 0th letter, and in the key, key[0] is W. So everywhere in my message I find an A, I write a W). Decrypting is the opposite process. You search for the target letter in the key, and convert the index of that letter into its alphabetic equivalent. (I found ‘Z’ at the second position in the key, so ‘Z’ must represent the letter ‘B’). In this problem, you are going to create a program to decrypt a message, given the key.

**Part A)** Using the prototype below, write a function called GetKey(). This function should use the "gets()" function in <stdio.h> to get a 26-letter string from the user and store that information in the array that is passed in. You can look up how to use gets() in your book.

Prototype: void GetKey( char array[27] ); /* get a 26-letter key from the user */

**Part B)** Using the prototype below, write a function called GetIndex. This function should search through an array of characters, and return the index where it finds the target. The array will be null-terminated for robustness, return a –1 if you do not find the target before you find the null character ‘\0’. The function should take an array of chars,

Prototype: int GetIndex( char target, char key[] );

example: if the key is [Z Y X W V U T S R Q P O N M L K J I H G F E D C B A]
GetIndex( D, key ) should return the index 22.

**Part C)** Using your GetIndex function from part b, write a function called Decrypt() that asks the user for a message, then goes through the message letter by letter and prints the decrypted version of the message. (You need to implement a search routine, without making use of functions supplied in the string.h library.)

Prototype: void Decrypt( char key[] );

**Part D)** Using the given main() function (see below) and the 3 functions you wrote in parts a-c, use your program to decrypt the following message. If you want to cut-and-paste the cyphertext into your program, you may find it (as well as a copy of the main() and Encrypt() functions within testbed.c) at http://web.mit.edu/16.070/www as message1.txt, message2.txt and testbed.c:

key = [D C B A H G F E L K J I P O N M T S R Q X W V U Z Y]

Message: <BSZMQNFSDEZLRDOHONSPNXRIZMNVHSQXIZQNIIQEDQOHARQNCHBNOQSN1IHAJKXRDVRHBNQOSNICNPQCTDOASNBHQR-ADWLAIDZQHI
MSHRLAHOQ'RNGGLBHNGRBLHOBHDOAQHBEONINFZMNILBZ .>
Turn In: Printout of all source code, screen dump of sample run, also copy your program as username_PS6_P2.c to %CDIO-Prime\16.070HW\ps6\p2%, replacing username with your Athena username. 
NB: You need to copy your functions into a file together with the given main() function so that we may run the code as one file. 
Indicate how much time you spent on this problem.

Problem 3 (40%)
The following problem should be considered a requirement specification for a software design problem.

A professor at a certain New England university wishes to perform some statistical analyses on her students’ exam grades. Design a program that allows her to input the students’ scores for the semester and then provides her with some useful statistics.

Your program should first ask the professor for the number of exams taken by the students and the number of students in the class. You may limit these quantities if you wish, but your program should be able to handle up to 20 exams and 100 students. You may assume every exam is taken by each student. Next, the program should provide an interface for the professor to enter the students’ grades.

The program should then compute and output an estimate of the mean and standard deviation of the grade distribution for each exam. Using those estimates, the program should compute and output an estimate of the correlation coefficients between the exams.

Use the data below to test your program. Note: These data were artificially generated.

<table>
<thead>
<tr>
<th>Student</th>
<th>Exam 1</th>
<th>Exam 2</th>
<th>Exam 3</th>
<th>Exam 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nirav</td>
<td>71</td>
<td>69</td>
<td>86</td>
<td>82</td>
</tr>
<tr>
<td>Louis</td>
<td>78</td>
<td>73</td>
<td>90</td>
<td>85</td>
</tr>
<tr>
<td>Lorraine</td>
<td>92</td>
<td>85</td>
<td>100</td>
<td>94</td>
</tr>
<tr>
<td>Thomas</td>
<td>73</td>
<td>73</td>
<td>93</td>
<td>89</td>
</tr>
<tr>
<td>Josh</td>
<td>79</td>
<td>71</td>
<td>87</td>
<td>83</td>
</tr>
</tbody>
</table>

Here is a possible sample run using the data above:

Number of exams (<= 20): 4
Number of students (<=100): 5
Enter grades for student 1: 71 69 86 82
Enter grades for student 2: 78 73 90 85
Enter grades for student 3: 92 85 100 94
Enter grades for student 4: 73 73 93 89
Enter grades for student 5: 79 71 87 83

Exam: 1 2 3 4
--------------------------------------
Mean: 78.60 74.20 91.20 86.60
Std Dev: 8.20 6.26 5.63 4.93
Corr w/ 1: 1.00 0.93 0.79 0.74
Corr w/ 2: 0.93 1.00 0.96 0.93
Corr w/ 3: 0.79 0.96 1.00 0.99
Corr w/ 4: 0.74 0.93 0.99 1.00

A few hints to help you as you complete the problem:

1. If you wish to use a 2-D array to store the data, be careful when passing it to functions. For more info see Section 11.8 in the C book.
2. You can use format specifiers to easily create even columns like in the output above. For more info see Section 7.5 in the C book.

3. You will need the mean to calculate the standard deviation. You will need both the mean and the standard deviation to find the correlation coefficient. Keep these facts in mind when writing your functions to calculate mean and standard deviation.

4. Some useful equations:

   For a set of N data-points:
   \[ X: \{ x_1, x_2, x_3, x_4 \ldots x_N \} \]

   The mean can be estimated as
   \[ \bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i \]

   The standard deviation can be estimated as
   \[ s_x = \sqrt{\frac{1}{N-1} \sum_{i=1}^{N} (x_i - \bar{x})^2} \]

   For a set of M sets of N data-points:
   \[ X_1: \{ x_{1,1}, x_{1,2}, x_{1,3}, x_{1,4} \ldots x_{1,N} \} \]
   \[ X_2: \{ x_{2,1}, x_{2,2}, x_{2,3}, x_{2,4} \ldots x_{2,N} \} \]
   \[ \vdots \]
   \[ X_M: \{ x_{M,1}, x_{M,2}, x_{M,3}, x_{M,4} \ldots x_{M,N} \} \]

   The correlation coefficient between data sets \( X_i \) and \( X_j \) can be estimated by:
   \[ r_{i,j} = \frac{\frac{1}{N-1} \sum_{k=1}^{N} (x_{i,k} - \bar{x}_i)(x_{j,k} - \bar{x}_j)}{s_{x_i} \cdot s_{x_j}} \]

**Turn In:** Printouts of the 5 steps of software design as detailed in Recitation 2. You may choose any one of the following three methods of implementing part 2 of step 3 (Design): Pseudo-code, Flow chart or State diagrams. Indicate the amount of time you spent working on this problem.

**Also:** Copy your program code as `username_PS6_P3.c` to `\CDIO-Prime\16.070HW\ps6\p3`, replacing `username` with your Athena username.