**III. Assignment**

Boolean Logic – Bboard Challenge

1. Challenge: Using only logical operators, set up the logic on the Bboard interface so that:   
    1) Bottom light: on if any number of the buttons are on (i.e., = 1)   
    2) Top light: on if and only if all three buttons are on   
    3) Middle light: on if any two buttons or more are on.

Turn in: A screen capture of the Bboard with this setup

1. Truth Table: Write a commented script to create a truth table for these cases in the Bboard Challenge. The truth table and Bboard should match.  
   Turn in: The script & the execution result.

Conditional Programs

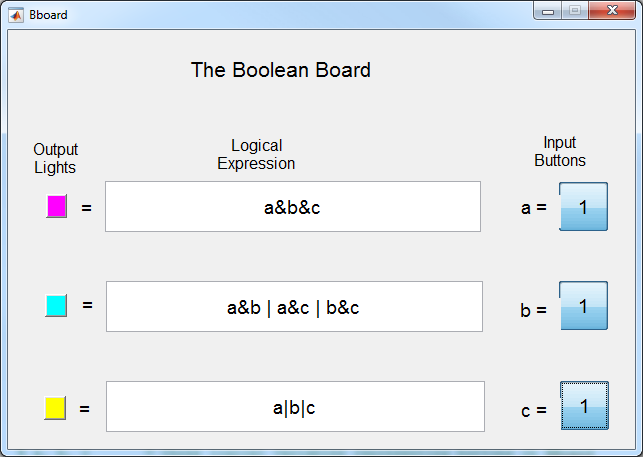
1. Make even Program (based on Figure 2)

Write a function that will carry out the logic in the Figure 2 flowchart for integer input.

Turn in: a. The flow chart for this program created I Visio using techniques from the Flowcharts Basic Tutorial available on the website. b. Complete commented code (step #5) and validation (step #6). Use provided test cases for the validation.

1. QC Program: Develop and Validate the QC prototype function outlined in section IV.3 above.  
   Turn in: A complete Function Development worksheet for the setup, coding, and validation

**Solutions**

1. Boolean Board Challenge 1: Setup the logic in the boxes so:   
    1) Bottom light: on if any number of the buttons are on (i.e., = 1)   
    2) Top light: on if and only if all three buttons are on   
    3) Middle light: on if any two buttons or more are on.   
   

Note: any logic which will meets the challenge requirements is acceptable.

1. Write a script to create a truth table for the middle case in Bboard Challenge 1.

**This solution does all three lights only the middle light is required.**

**Script**

%% Program Bchallenge1.m

% This script prepares a truth table for Boolean Challenge 1

%

% Scott Moor March 2016

% Variables

% a, b, c = three logical variables representing buttons on BBoard

% L1, L2, L3 = conditional logic for Lights 1,2 and 3 respectively

% Ttable = the resulting truth table

%% Set up input variables, Buttons 1, 2 and 3

a = [1 1 1 1 0 0 0 0]';

b = [1 1 0 0 1 1 0 0]';

c = [1 0 1 0 1 0 1 0]';

% Logic Calculations for Lights 1, 2 and 3

L1 = a & b & c;

L2 = (a&b) | (b&c) | (a&c);

L3 = a | b | c;

%% Calculate and display Truth Table

disp('Truth Table for Boolean Challenge 2')

disp('Variable order: Buttons 1, 2 and 3, Lights 1, 2 and 3')

Ttable = [a b c L1 L2 L3]

**Execution**

>> Bchallenge1

Truth Table for Boolean Challenge 2

Variable order: Buttons 1, 2 and 3, Lights 1, 2 and 3

Ttable =

1 1 1 1 1 1

1 1 0 0 1 1

1 0 1 0 1 1

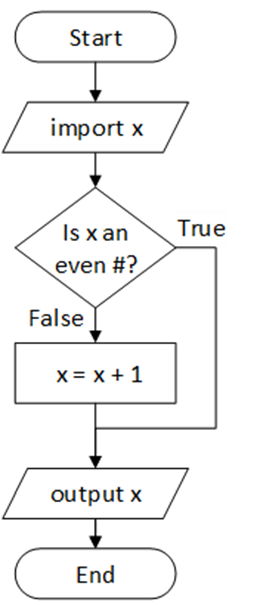
1 0 0 0 0 1

0 1 1 0 1 1

0 1 0 0 0 1

0 0 1 0 0 1

0 0 0 0 0 0

1. Make even Program (based on Figure 2) Write a function that will carry out the logic in the Figure 2 flowchart for integer input.

Turn in: a) The flow chart for this program created I Visio using techniques from the Flowcharts Basic Tutorial available on the website. 🡺

b) Complete commented code (step #5) and validation (step #6). Use provided test cases for the validation.

Function Code:

function x = MakeEven(x)

% This function forces an integer input to be even by increasing any odd

% number by one.

% S. Scott Moor March 2015

%

% function x = MakeEven(x)

% input: x = any integer

% output: x = an even integer (same as input or one higher)

% Conditional tests if the input is even and adds one if it is not

if rem(x,2) ~= 0

x = x + 1;

end

|  |  |
| --- | --- |
| **Input** | **Expected Output** |
| 2 | 2 |
| 3 | 4 |
| -4 | -4 |
| -3 | -2 |

Validation:

>> MakeEven(2)

ans =

2

>> MakeEven(3)

ans =

4

>> MakeEven(-4)

ans =

Matches expected results (shown above)

-4

>> MakeEven(-3)

ans =  
 -2

1. QC Program: Develop and Validate the QC prototype function outlined in section II.3 above.  
   Turn in: A complete a Function Development worksheet for the setup, coding and validation

Problem ID \_\_\_QC Function Programmer \_\_\_S. Scott Moor \_\_ \_\_\_\_

Set Up/ Planning Type of Program:  Script 🗹 Function

1. Problem Statement:

Create a simple program to detect if a part is within the QC specification for diameter (9.95 to 10.05 mm). Program is to return a 1 if the part is between the specified limits and a 0 if it is outside those limits.

1. Inputs: (full name, variable to be used, units)

|  |  |  |  |
| --- | --- | --- | --- |
| Variable Name | Description | Units or Values | Input Source\* |
| d | Diameter of the part | mm | Command line |

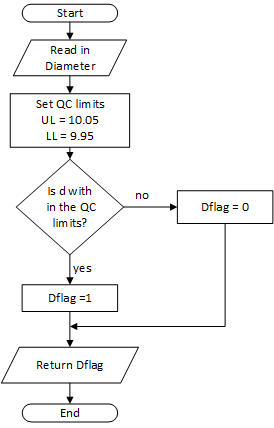
1. Output: (full name, variable to be used, units)

|  |  |  |  |
| --- | --- | --- | --- |
| Variable Name | Description | Units or Values | Output type\* |
| Qflag | Quality flag (1 for in spec.) | 1 or 0 | Command line |

1. Solution Steps (order of these two parts may be varied):
2. Perform calculation on test case(s) (2) Identify the steps/equations to be used in code

(select a test case if one is not given, avoid answers of 0 or 1)

1. Test Cases: (2) Steps:

Diameter (mm) Result (1 = in spec)

9.96 1

9.94 0

10.5 1

10.7 0

5. Code *(I usually cut, paste and edit for this page)*

Note: Program steps will most likely be presented as a list instead of a flow chart. This is fine (we have barely introduced flow charts at this point). However, the flow chart is ideal and some description of steps is expected.

% This function will test if a part diameter is with in the needed

% specification limits. Limits can be adjusted in the program

% Scott Moor March 2016

%

% function Dflag = QCdiameter(d)

% Input Variable: d = the diameter of the part (mm)

% Output Variable: Dflag = an output flag: 1 = in spec, 0 = out of spec.

% Internal variables

% UL = Upper Limit on the quality specification (mm)

% LL = Lower Limit on the quality specification (mm)

function Dflag = QCdiameter(d)

% Set Quality Limits

UL = 10.05;

LL = 9.95;

% Determine if value is with in the limits and return appropriate flag.

if d>=LL && d<= UL

Dflag = 1;

else

Dflag = 0;

end

6. Validation:   
Prove that the function is working correctly. Generally the function should be run on the test case from step 4 and on a wider range of cases.

>> QCdiameter(9.96)

These cases match earlier test calculations and cover values that are high, low and in spec

ans =

1

>> QCdiameter(9.94)

ans =

0

>> QCdiameter(10.05)

ans =

1

>> QCdiameter(10.07)

ans =

0