

# CS344 Lab 3 – Trapezoid II

Assigned: 9/11/09

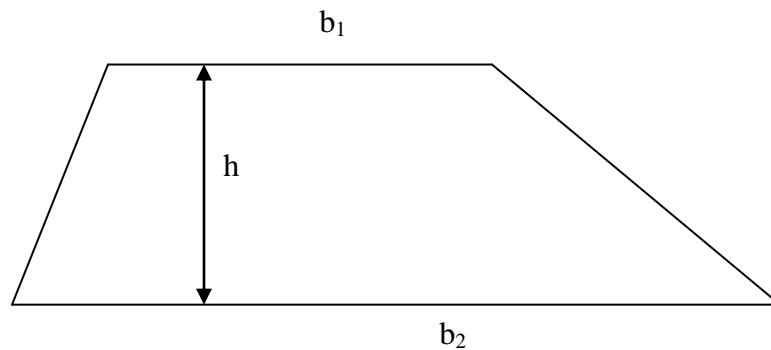
Points: 20

## Purpose

The purpose of this lab is to understand how to check the input value in FORTRAN and to perform similar actions in MATLAB.

## Process

In this lab we will prompt the user for the base and height of a trapezoid and will then display its area.



The area of a trapezoid is

## FORTRAN Implementation

Modify Lab 2 such that it tests for valid numbers. A valid number is considered to be a number greater than 0. You will need to use the *if* statement to perform this task. Your output should be similar to the following:

```
Enter b1
0
Bad value - Use a value greater than 0

Enter b1
-4
Bad value - Use a value greater than 0

Enter b1
```

```
4
Enter b2
-3
Bad value - Use a value greater then 0

Enter b2
3

Enter height
0
Bad value - Use a value greater then 0

Enter height
4
The area of the rectangle is      14.00
```

## MATLAB Implementation

Compute the area of a trapezoid using MATLAB. The following examples will equip you with the basic knowledge to complete the lab.

### Input

To read a value use the *input* function. The string in the function is displayed and the return value is assigned to the variable to the left of the assignment operator. Notice that strings are enclosed with single quotes in MATLAB.

```
>> num = input('Enter a number: ');
Enter a number: 34
```

### Output

To display information, use either the *display* function or the *fprintf* function. The *display* function is simpler but not as powerful. It takes a single argument that is written to the screen.

```
>> display('A simple string');
A simple string
>> display(area);
```

```
area =
```

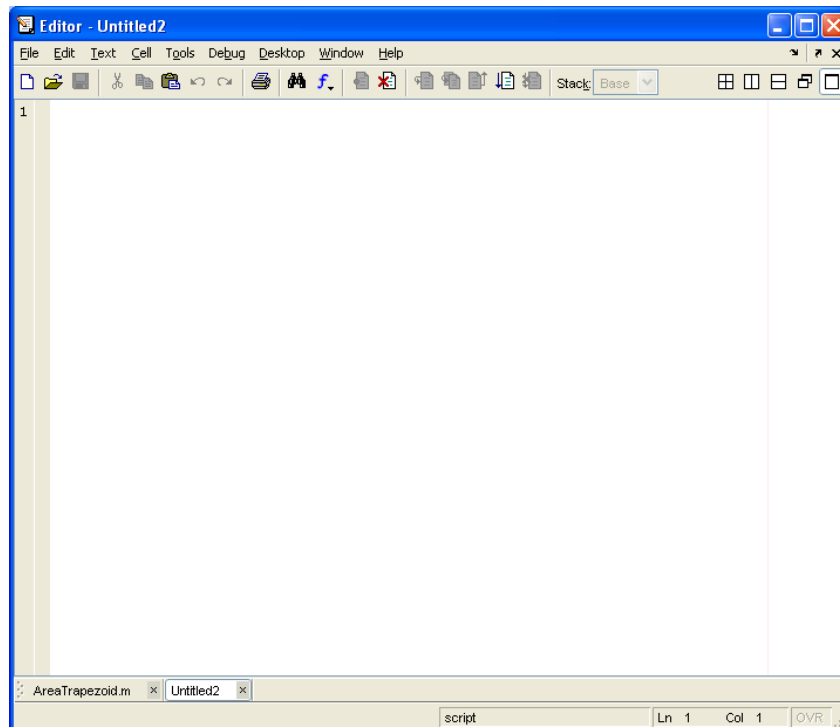
```
10
```

The *fprintf* function is more complicated but is more powerful. It is similar to the *printf* statement in C. The simplest form consists of a string followed by a variable to be displayed. Within the string is text and a format field. In the example below, the format field is %f. The variable sum is displayed in the %f field. Notice that 8.4 precedes the field type specifier and works similar to the way it works in FORTRAN. The \n will cause a new line to be printed.

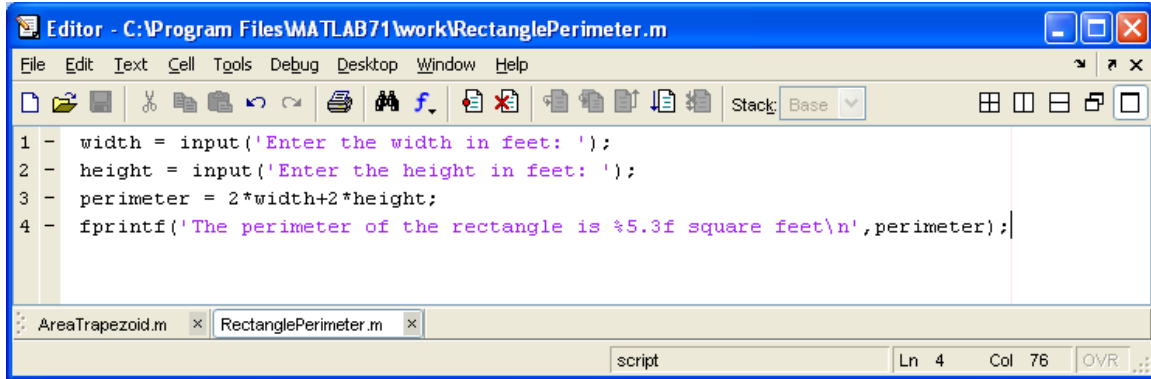
```
>> sum = 56;  
>> fprintf('The sum is %8.4f\n',sum);  
The sum is 56.0000
```

## M-Files

You can put all of the code used to perform a computation in an .m file. To create a .m file use the File - New - M-File menu command. This will bring up an editor window.



Enter the code to be executed in the window.



The screenshot shows a MATLAB Editor window titled "Editor - C:\Program Files\MATLAB71\work\RectanglePerimeter.m". The window contains a script with the following code:

```
1 - width = input('Enter the width in feet: ');  
2 - height = input('Enter the height in feet: ');  
3 - perimeter = 2*width+2*height;  
4 - fprintf('The perimeter of the rectangle is %5.3f square feet\n',perimeter);
```

The window also shows a toolbar with various editing and debugging tools, and a status bar at the bottom indicating the current line and column (Ln 4, Col 76).

Save the file using a meaningful name such as *RectanglePerimeter*.

At the MATLAB command prompt, enter the name of the M-file to be executed and the file will be executed.

```
>> RectanglePerimeter  
Enter the width in feet: 2  
Enter the height in feet: 5  
The perimeter of the rectangle is 14.000 square feet
```

You will need to create a similar M-file for this lab. The output of your program should be similar to the following:

```
>> AreaTrapezoid  
Enter b1: 7  
Enter b2: 3  
Enter height: 7  
The area is 35.00
```

Turn in a single Word document with the code and sample output for both of the three implementations.