Massachusetts Institute of Technology
Department of Aeronautics and Astronautics
Cambridge, MA 02139

Unified Engineering
Spring 2005

Problem Set #8

Due Date: Tuesday, April 5, 2005 at 5pm

	Time Spent (minutes)
S5	
S6	
C5	
C6	
C7	
Study Time	

Name: ________________

Problem S5 (Signals and Systems)

Consider the signals

$$g(t) = \begin{cases} 0, & t < 0 \\ e^{-2t}, & t \geq 0 \end{cases}$$

and

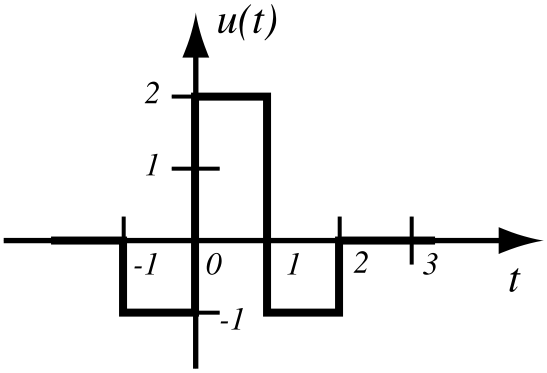
$$u(t) = \begin{cases} 0, & t < -1 \\ e^{-t}, & -1 \leq t \leq 1 \\ 0, & t > 1 \end{cases}$$

1. Sketch the two signals. Your sketch should be fairly accurate, as this will aid in the rest of the problem.

2. Using graphical convolution techniques, sketch the signal $y(t) = g(t) * u(t)$. Make sure to label important features on the graph, such as points in time where the graph of $y(t)$ changes character, the value of $y(t)$ at those breakpoints, etc. The goal of this part is not to calculate $y(t)$ exactly, but to rather to use graphical convolution to determine (approximately) what the final solution should look like. It's always a good thing to know the answer before you have to calculate it!

3. Now evaluate the convolution exactly. You should be able to use your results in part 2 above to help identify the limits of convolution.

4. A different signal $u(t)$ is shown in the graph below. Sketch, as accurately as possible, the signal $h(t) = u(t) * u(t)$. Explain your reasoning.



Problem S6 (Signals and Systems)

Note: This problem is similar to one given a couple years ago. Please try to do this one without looking at bibles — the solution is instructive.This problem shows, in a somewhat simplified way, how a radar system determines the distance to an aircraft. More basically, it will give you practice doing convolution using the "flip and slide" method.

The radar sends out a signal, $u(t)$, that reflects off the aircraft and returns to the radar system. The time it takes the signal to return is twice the distance to the aircraft, divided by the speed of light. The received signal is $u(t-T)$, where T is the round trip travel time of the signal. It may not seem obvious, but it's not easy to determine T by measuring the returned signal. In principle, one could simply display the returned signal on an oscilloscope and measure the time of some key feature of the signal, such as the leading edge of the signal, or the peak of the signal, or whatever. However, the signal received by the radar is relatively weak, because most of the energy radiated by the radar does not strike the aircraft, and most of the reflected energy is not received by the radar. Often the noise in the received signal is as large or larger than the signal itself.

Instead, a better approach is to try to identify the whole received signal instead of a single key feature. How can we do this? Convolution! The received signal is run through an LTI system G with impulse response $g(t)$. The impulse response is designed to accentuate the signal, so that one can measure a key feature of the output signal, $y(t)$, which is relatively insensitive to noise. Often, the best choice for $g(t)$ is a time-reversed version of $u(t)$. That is,

$$g(t) = u(-t) \tag{1}$$

This is called a "matched filter."

1. For the signal $u(t)$ shown below, and $g(t) = u(-t)$, find the convolution

   $$y(t) = g(t) * u(t)$$

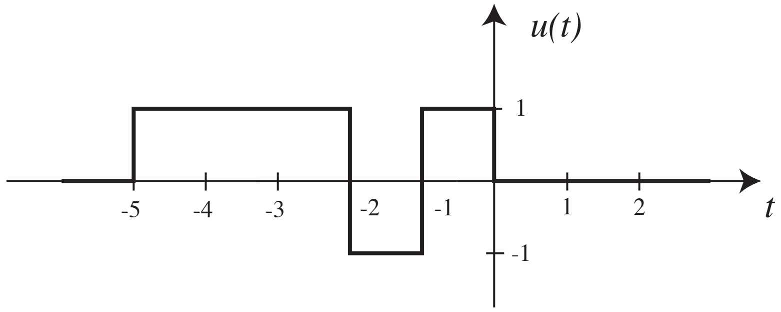
   using the flip and slide method.

2. Using the results of part (1) and the LTI properties of the system G, what is the convolution

   $$y(t) = g(t) * u(t-T)$$

3. What feature of $y(t)$ would you use to identify the time T?

4. Can you explain why the right impulse response for G is a signal that has the same shape as $u(t)$, but time reversed?


u(t)
1
-1
-5
-4
-3
-2
-1
1
2
t

The problems in this problem set cover lectures C5, C6, and C7.

Download and unzip the following file from the C&P class webpage:

`C&P_PSet2_Files.zip`

Some answers (C6c, C7a, C7b) will need to have their answers zipped and uploaded to:

https://spacestation.mit.edu/unified/

Two *modified* files are expected:

`expression_tree.ads` AND `expression_tree.adb`

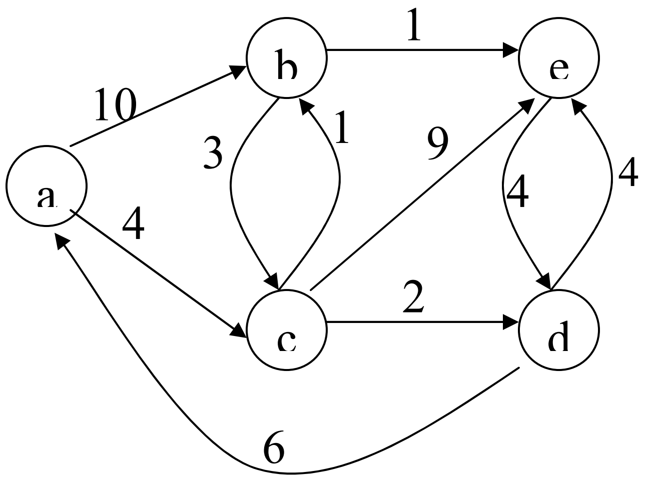
Please zip/compress/store these two .adb files. Name the file:

"C67_Lastname_Firstname.zip"

Problem C5. Graphs, Shortest Path

What is the shortest path from the vertices **a** to all the other vertices in the graph shown below?

Show all the steps in the computation of the shortest path, using Dijkstra's algorithm.



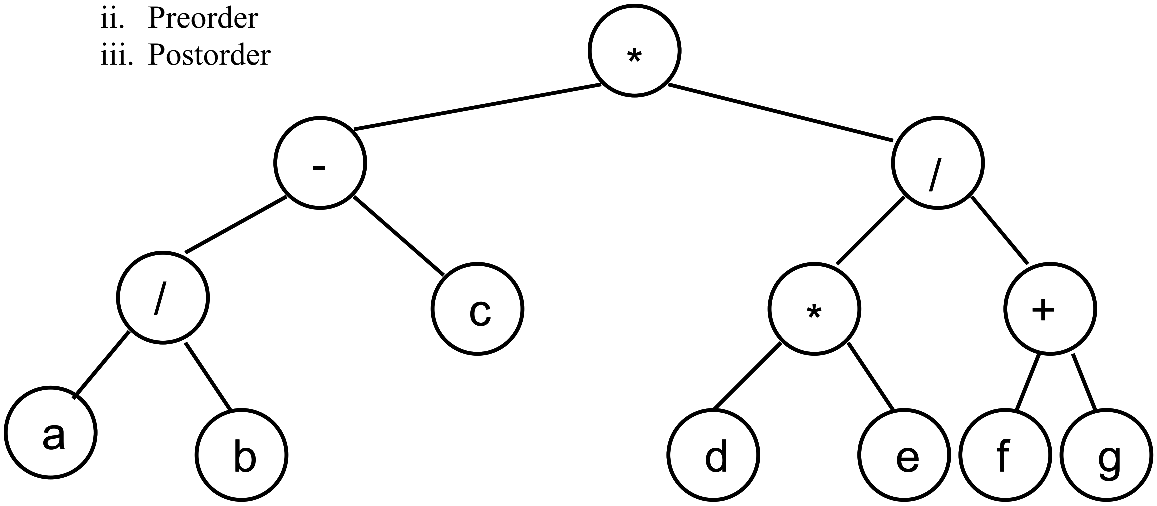
Problem C6. Tree Traversal

With the following files:

- ❖ `my_pointer_stack.ads`
- ❖ `my_pointer_stack.adb`
- ❖ `expression_tree.adb`
- ❖ `expression_tree.ads`
- ❖ `test_expression_tree.adb`

a. Given the tree below, traverse the tree

i. Inorder
ii. Preorder
iii. Postorder



It is sufficient to show the final expression you obtain after traversing the tree. For example, the start of the answer for **a.** 'Inorder' traversal would look like:
"a / b – c "…

b. Write an algorithm (pseudo code) to create a binary expression tree using an input string in prefix notation. *(hint: look at what the code does currently)*

c. Modify the `Create_Expression_Tree` procedure in the file `expression_tree.adb` to use a prefix expression as the input string. (In this implementation you do not need to worry about handling negative numbers or numbers with more than one digit)

Note: You can still use `test_expression_tree.adb` to check the correctness of your answer.

Turn in a hard copy for part a, part b (your algorithm/pseudo_code). Part c., `expression_tree.adb` will be modified again for Problem C7. Wait to print out a hardcopy code listing and submit an electronic copy online until you have completed C7.

Problem C7. Spanning Trees/ Depth-First and Breadth-First Traversal

With the same files from C6...

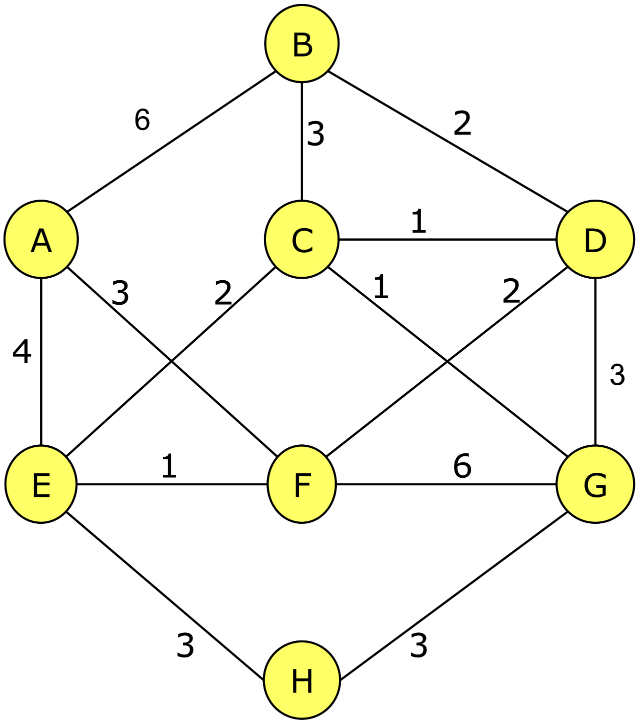
a. Modify `expression_tree.ads` to include headers for procedures:
   1) Traverse_DepthFirst (depth-first traversal)
   2) Traverse_BreadthFirst (breadth-first traversal)

b. Modify `expression_tree.adb` to include procedures for depth-first and breadth-first traversal (You already modified it in C6, modify it again).

   These procedures should behave similarly to the 'Traverse_...' procedures already given (i.e. *recursively* display the value at the node followed by a space as the traversal is happening)

Note: You can use `test_expression_tree.adb` to check the correctness of your answer by uncommenting the marked lines.

c. Find the minimum spanning tree, using Prim's algorithm, for the graph shown below. Illustrate all the steps in the creation of the minimum spanning tree



Turn in a hardcopy code listing of the modified `expression_tree.ads` AND `expression_tree.adb` along with a hardcopy of your answer to part c. Submit an electronic copy of `expression_tree.ads` AND `expression_tree.adb` online.