remedial software engineering kick-off!

ESE2025

here we feature a series of exercises to get started!

1. find the largest integer from a given array of integers
    a. write the pseudocode first (input: an integer array (of unique integers); output: an integer)
    b. run some thought experiments on your pseudocode, try regular cases and extreme cases (what if the array is empty? what if some of the integers are the same?)
    c. once you are confident, write your algorithm in C syntax
    d. next, write your algorithm as a proper C function, passing in the array using a pointer; what do you also need to pass in to your C function?
2. bubble sort an array of integers
    a. write the pseudocode
    b. try some cases
    c. convert to C
    d. write the function

notes on pseudocode

- don’t just admire it! use it…
- you use it to anticipate deficiencies in your logic (and any resulting software)
- cases to consider:
  - regular cases (nominal situations, relatively easy to think of)
  - extreme cases (at the limits of sizes and inputs, for example)
  - absurd cases (when things are not OK outside the programming context)

getting started with Git

- please have a look at the Git material in Derek Molloy's Exploring Beaglebone (second edition), starting on page 124 (it's a solid introduction and brief)
- Git is a version control program, used for collaborating
- virtually any kind of file can be supported, from plain language documents, graphics projects, PCB layouts or software
- Git was developed by Linus Torvalds as a means of facilitating collaboration on the Linux kernel (involving thousands of developers across a number of continents!)
- Git is a program that runs on your machine; to use it, you will need access to a *repository*; for our purposes, we will use github.com for our repositories

git (cont'd)

- we recommend creating new repositories on github.com itself (it is a simple procedure--- just follow the directions on github.com)
- once the repository is created, you can *clone* to your personal machine, via the git clone command:
    - $ git clone <repository_name>
- where "$" denotes the Linux prompt.
- the repository name can be URL for the repository. For example, by teaching repository has the URL: https://github.com/takiszourntos/teaching
- sometimes the ".git" extension is added to the repository name

git (cont’d)

- once cloned, you have a local version of the github.com repository, sometimes referred to as the *local repository*.
- you now can work with the local repository and make changes to it (often it corresponds to an Eclipse or MCUXpresso workspace for this class)
- when you have made your changes, you can send your changes back up to the github.com repository
- to do this, first add your changes:
    - $ git add .
- this places your changes in a “staging area”
- then, commit your changes to the local repository:

more git!

- to do this, enter:
  - $ git commit -m “place a meaningful but short comment here, for the record”
- and, finally, to push up your changes to the remote repository we use a push:
  - $ git push
- which will require you to enter your github.com username and password
- for the changes to count towards your “green squares” tally, you need to make sure you “register” your local machine:
  - $ git config --global user.name “Takis Zourntos”
    - (use your own name instead!)
  - $ git config --global user.email “takis.zourntos@emads.org”
    - (again, use your own email address, the same one you used to register your github.com account!)