Programs written in any language have the same basic elements:

- Input and output information
- Store information
- Process information, e.g., add two numbers
- Make decisions and control order of execution
- Repeat instructions
- Procedures - call a set of instructions with a single instruction

Matlab uses ***functions*** for procedures. Other languages also use *subroutines* and some use *objects*.

These few elements can be combined together in order to create programs with complex behavior.

The complex behavior results because these program elements can be used in combination with each other:

- Decision and repeat structures can be nested inside of each other.

- A procedure can call other procedures and some can call themselves.

With only these few basic elements, when used in combination with input and output devices, computers can do all sorts of amazing things!

Richard K. Herz, ReactorLab.net

With only these few basic elements, when used in combination with input and output devices, computers can do all sorts of amazing things!

This is a VERY IMPORTANT IDEA:

The interactions between only a few simple processes can produce very complex behavior!

For example, the genetic code is recorded in the sequence of only 4 types of chemical groups along DNA molecules but can be expressed as a plant, animal, or human being!

> *Hmmm, digital computers store data as a string of two numbers (0,1) in a binary number (base-2) system. A DNA molecule stores data for a living organism as a string of four chemical groups (abbreviated A,G,C,T), or we can think of them as four numbers (0,1,2,3) in a quaternary number (base-4) system. Can you think of any advantages to the quaternary system used in DNA over the binary system used by digital computers?*

The interactions between only a few simple processes can produce very complex behavior!

This is true in many physical systems, e.g., fluid flow, biological systems, chemical reactions, mechanical systems. It only takes a couple phenomena that interact with each other to create very complex behavior.

Of course, an implication of this is that a simple mistake, e.g., an instruction in the wrong place, can lead to a big problem, e.g., your robot crashes into the wall or an insect gets an extra pair of legs.

Richard K. Herz, ReactorLab.net

The easy part about computer programing is learning a relatively small list of instructions.

The hard part is putting these instructions together in such a way that the computer accomplishes your goal.

Thus, the real challenges are in problem definition, logic, and problem solving.

Richard K. Herz, ReactorLab.net

Your tasks as a computer programmer:

- Design, write, test and correct a program that makes the computer accomplish your goal.

- Each of these steps - design, write, test and correct - will be performed repeatedly in the process of producing your final program.

- The computer always does what your program tells it to do. Your challenge is to make your program tell the computer to do the correct things in the correct order to accomplish your goal.

- Write a program that is easy to read and understand by other people and by yourself some time after you finish writing it. You need other people to understand your work in case they need to modify it or extend it (or give you a grade in this course!).

  This includes adding comments, separating data from instructions, logical structuring of code, and breaking the work into procedures so each section of your work is relatively short and easy to read and de-bug.