

# **Algorithmic Thinking**

**Computers are not people!**

**People can think on their own.**

**Computers must be told exactly what do!**

**Computers must be told the exact sequence of instructions to follow in order to accomplish a task.**

## Programming to Solve a Problem: (Software Life-Cycle)

1. Determine an **algorithm** to solve the problem:  
An algorithm is a sequence of steps that can be followed to solve the problem. (An algorithm is like a recipe.)
2. **Write a program** to implement the algorithm:
  - use very specific, detailed instructions.
  - use a “language” the computer understands.
3. **Compile** the program:  
A compiler translates the program into machine language. Fix any syntax errors that the compiler finds.
4. **Run (or execute)** the program:  
At this time we may find other errors (in execution or logic) that must be corrected.
5. **Debug** the program:  
Test the program and fix any errors.
6. **Maintain** the program:  
Keep monitoring the program for unexpected errors.

Note: An **Interpreter** translates a program to machine language as it is being loaded and executed.

## Examples of Algorithms:

- **Find the oldest person in the room:**

- Go through the people sequentially, keeping track of the oldest so far.
- have the people pair up, eliminate the younger of each pair. Keep pairing up and eliminating the younger ones of the pair until there is only one left.

- **Search for telephone number:**

- Sequential search
- Binary search

## Examples of Bugs:

1. The Mars Climate Orbiter was lost in space in 1999. In the calculation of its route, some programmers used the metric system while others used English units!
2. A Patriot missile failed to intercept a scud fired at US troops in 1991. The following is from an article on the incident “Specifically, the time in tenths of a second, as measured by the system’s internal clock, was multiplied by 1/10 to produce the time in seconds.”

A **recipe** is a good analogy of a program:

- ingredients
  - constants, variables
  - data types, integer and real numbers, characters
- detailed step-by-step-instructions
  - statements
- repeated actions (e.g., separate 5 eggs)
  - loops
- pre defined actions (e.g., saute, puree, etc.)
  - functions, parameters
- decisions (e.g., bake until brown, beat until firm, etc.)
  - conditions