

## Solving Problems using a Computer

Methodologies for creating computer programs that perform a desired function.

Problem Solving

- How do we figure out what to tell the computer to do?
- Convert problem statement into algorithm, using stepwise refinement.
- Convert algorithm into LC-3 machine instructions.

Debugging

- How do we figure out why it didn't work?
- Examining registers and memory, setting breakpoints, etc.

Time spent on the first can reduce time spent on the second!



## Stepwise Refinement

Also known as systematic decomposition.

Start with problem statement:
"We wish to count the number of occurrences of a character in a file. The character in question is to be input from the keyboard; the result is to be displayed on the monitor."

Decompose task into a few simpler subtasks.

Decompose each subtask into smaller subtasks, and these into even smaller subtasks, etc.... until you get to the machine instruction level.

## Problem Statement

Because problem statements are written in English, they are sometimes ambiguous and/or incomplete.

- Where is "file" located? How big is it, or how do I know when I've reached the end?
- How should final count be printed? A decimal number?
- If the character is a letter, should I count both
upper-case and lower-case occurrences?


## How do you resolve these issues?

- Ask the person who wants the problem solved, or
- Make a decision and document it.




## LC-3 Control Instructions

How do we use LC-3 instructions to encode the three basic constructs?

## Sequential

- Instructions naturally flow from one to the next,
so no special instruction needed to go
from one sequential subtask to the next.

Conditional and Iterative

- Create code that converts condition into $\mathbf{N}, \mathbf{Z}$, or $\mathbf{P}$.

Example:
Condition: "Is R0 = R1?"
Code: Subtract R1 from R0; if equal, $Z$ bit will be set.

- Then use BR instruction to transfer control to the proper subtask.


Problem Solving Skills
Learn to convert problem statement into step-by-step description of subtasks.

- Like a puzzle, or a "word problem" from grammar school math.
$>$ What is the starting state of the system?
$>$ What is the desired ending state?
$>$ How do we move from one state to another?
- Recognize English words that correlate to three basic constructs:
$>$ "do A then do B " $\Rightarrow$ sequential
$>$ "if G , then do H " $\Rightarrow$ conditional
$>$ "for each $X$, do $Y$ " $\Rightarrow$ iterative
$>$ "do Z until W" $\Rightarrow$ iterative



The Last Step: LC-3 Instructions Use comments to separate into modules and to document your code.



```
Summary
Creating a machine program
- Convert problem statement to algorithm
- Convert algorithm to machine code
Stepwise refinement
- Sequential construct
- Conditional construct
- Iterative construct
Mapping to LC-3 instructions
```

