Problem solving using systematic decomposition

**Programming in LC-3 Machine Language - Part I**

**Systematic decomposition**

To solve a problem we need to go through several levels of transformation:

<table>
<thead>
<tr>
<th>Problem statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithm</td>
</tr>
<tr>
<td>Program</td>
</tr>
</tbody>
</table>

\[ \downarrow \]

In systematic decomposition, complex tasks are broken down into simpler tasks.
LC-3 instructions to implement constructs

* Sequential construct:
  Subtasks are executed one after the other

Flowchart:

```
  Subtask 1
  │
  │
  │
  Subtask 2
  ```
Example: \( R_4 \leftarrow R_1 + R_2 + R_3 \)
**Conditional construct:**

- Use condition codes (N, Z, P) to test
- Use conditional branch (BR) to choose subtask to execute
- Use unconditional branch to transfer control after finishing subtask 1
Example: $R_2 \leftarrow |R_1|$
* Iterative construct:

Flow chart:

1. Use condition code \((N,T,P)\) to test
2. Use conditional branch (BR) to finish iterating
3. Use unconditional branch to test again
Example:

\[
R1 \leftarrow 0
\]

\[
\text{while } (R1 < 4)
\]

\[
R1 \leftarrow R1 + 1
\]