

Computational Thinking

Decomposition

Divide and Conquer

One of the traditional ways to solve a problem



Break it into smaller problems



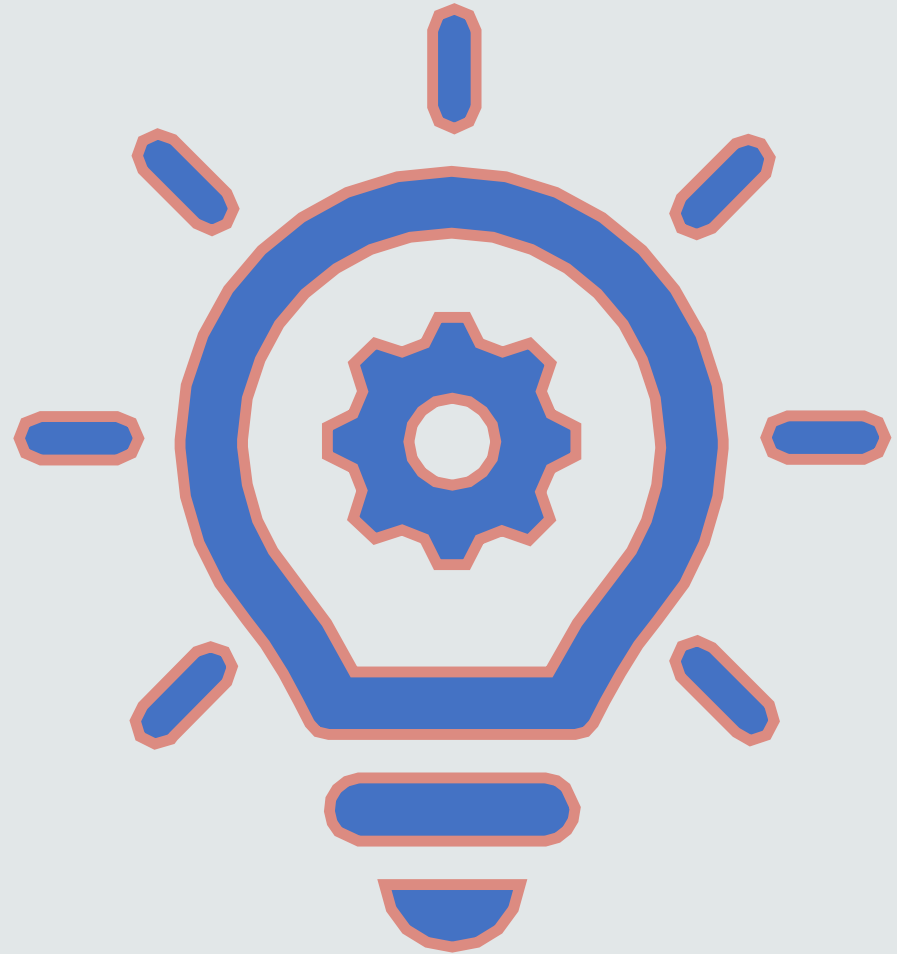
Solve each of the smaller problems



Combine the solutions to the smaller problems
to yield a solution to the whole problem

Example One – Summing several values

- Analyze the problem
- We need a place to store the sum
- The sum needs to be initialized to zero
- We need to access each of the values
- We need to add each of the values onto the sum



Decompose the problem

Create and initialize sum

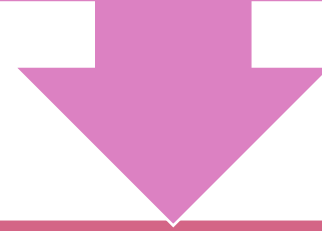
Access each value in turn by

- *having user enter the values one at a time*

Add each value onto the sum

*Example
Two –
Sort two
values into
ascending
order*

Compare the first
value to the second



If they are out of
order, swap them

*But,
swapping
is not as
easy as it
looks*


If we copy the second value to the first

We overwrite and lose the first value

We need to come up with a way of swapping that will preserve the first value

Swapping

We can use a third variable to store the first value when we overwrite with the second



Copy the first value to temp



Copy the second value to first



Copy temp to the second value



Putting it all together



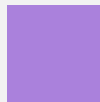
Compare first to second



If they are out of order



Copy first to temp



Copy second to first



Copy temp to second