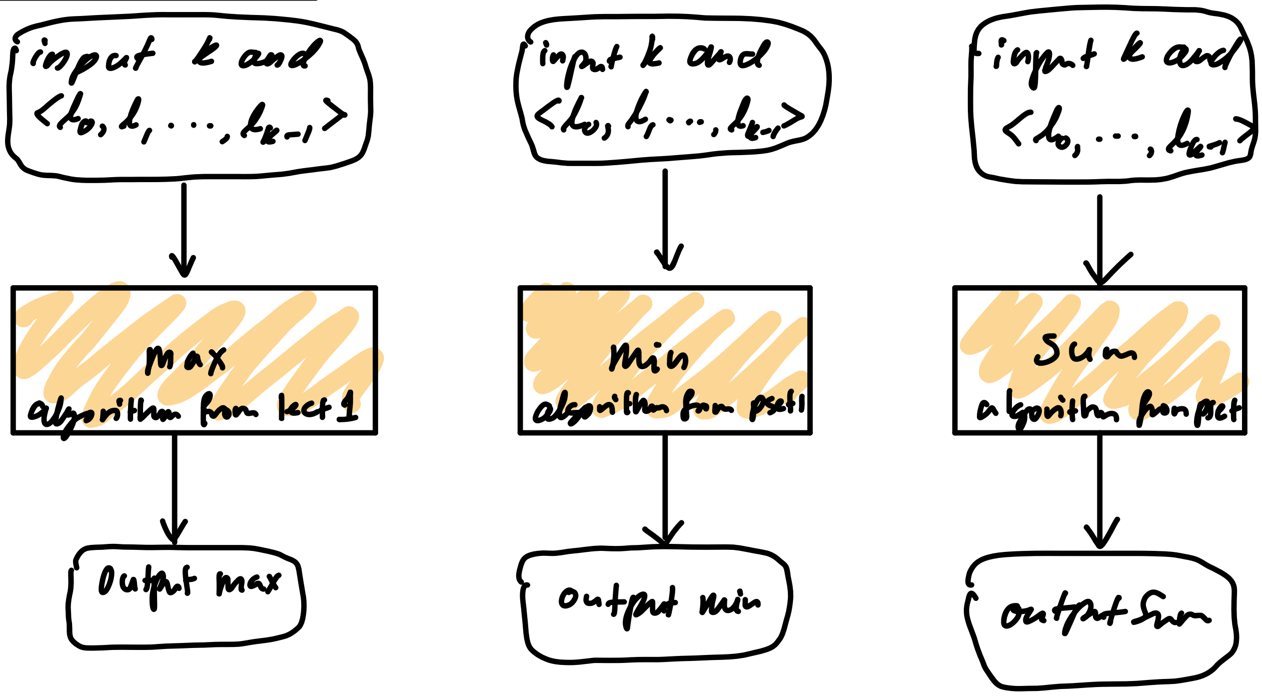
Functions and Types

Unit 3: Functions



Example: Computing range given a set of values
Decomposing the problem:
Range requires the (i) max of a set, (ii) min of a set and (iii) difference between max and min.

$$\text{Range} = \max(L, k) - \min(L, k)$$

Dividing the problem into simple subproblems
→ Wishful thinking - Assume that we know the solution of subproblems. Focus on what the function does rather than how the function does it.

Example 2: Find the mean of a set
Subproblems: (i) Sum of all values, (ii) number of elements
sum(L) len(L)

$$\text{mean}(L) = \frac{\text{sum}(L)}{\text{len}(L)}$$

Example 3: Find the standard deviation within a given set

$$\sigma = \sqrt{\frac{\sum_{i=0}^{L-1} (l_i - \mu)^2}{K}}$$

Subproblems:
- mean function
- subtract function (decrease every value in the set by mean)
- square function: computes squares
- sum function
- len function
- square root function

set μ to mean(L)
set L' to subtract(L, μ)
set L'' to square(L')
set total to sum(L'')

total = sum(square(subtract(L, mean(L))))

To find the standard deviation, we find the mean of L'' and apply a square root to it.

std_dev = sqrt(mean(square(subtract(L, mean(L)))))

On solving higher-level problem we solve the lower-level subproblems.

THE PROGRAMMER'S MANTRA

break down a problem into sub-problems and solve them one-by-one with functions. Compose these functions to solve the original task

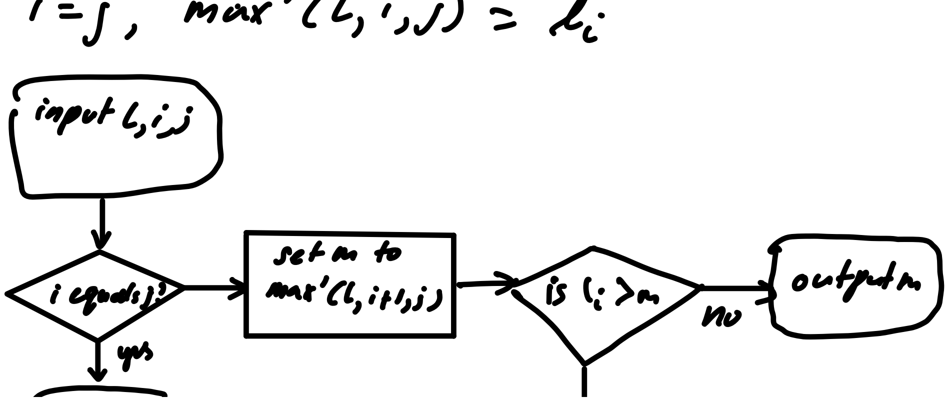
Alternative approaches to maximum:

Assuming I can only find max(L, k) for a small set of numbers:

Let max'(L, i, j) return the maximum among $l_i, l_{i+1}, \ldots l_j$

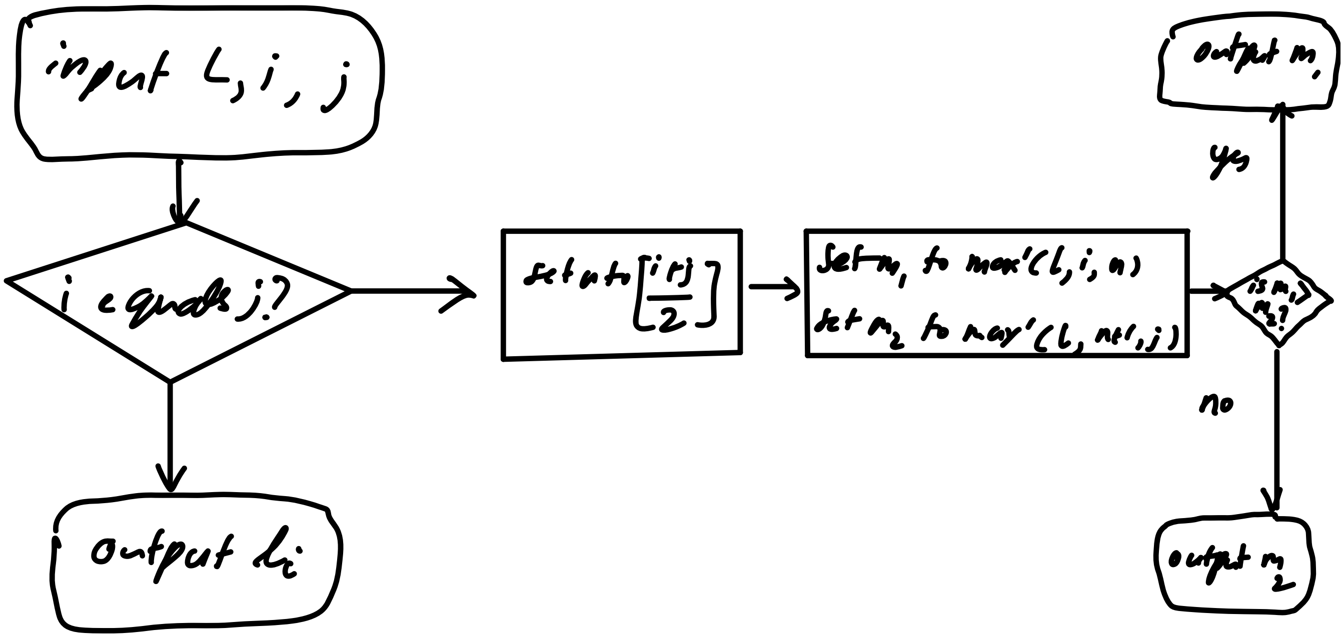
$$\max(L, k) = \max'(L, 0, k-1)$$

if $i = j$, $\max'(L, i, j) = l_i$



Consider splitting the list into two lists.
Given $max'(L, i, j)$, let $n = [(i+j)]/2$ we can solve $max'(L, i, j)$ by solving:

$max'(L, i, n)$ and $max'(L, n+1, j)$



Functions that call themselves - Recursive functions
E.g., factorial, max'

factorial(n) = n * factorial(n-1)

Types

A type is represented by a fixed, finite number of bits

1 bit : 1 or 0 can use it to represent 2 different values (e.g., true or false)

2 bits : 11, 10, 01, 00 can use it to represent 4 different values

For k bits, we can represent 2^k values.
more bits requires greater memory (can represent bigger range)
fewer bits require less memory (but represents smaller range)

8 bits : $2^8 = 256$ integers

0 to 255 (for unsigned) ← only represents non-negative int
-128 to 127 (for signed) ← represents both +ve and -ve int

8 bits : 127 different symbols using ASCII standards

Infinite real numbers but finite number of bits. Results in issues when processing floating point numbers.

Variables must be declared with the correct type to produce the correct output.

C is a static typed language (once type is declared, it cannot be changed).

Homework:

d

(1) Problem Set in Unit 3

(2) ssh into CS1010 PE hosts and follow the Unix CLI an vim tutorial

(3) Diagnostic Quiz for Lecture 2 (due on Weds)

(4) Diagnostic Quiz for PE (due next Fri)