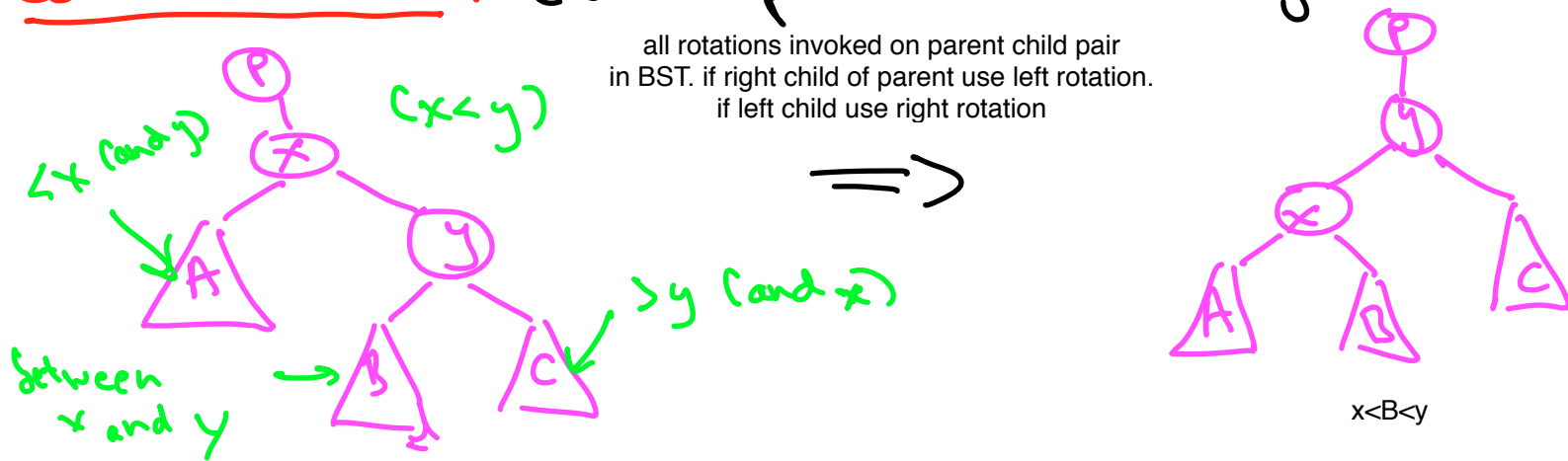# Data Structures

Rotations

Design and Analysis
of Algorithms I

Left Rotations

key primitive: rotations. (common to all balanced search tree implementations — red-black, AVL, B+, etc.)
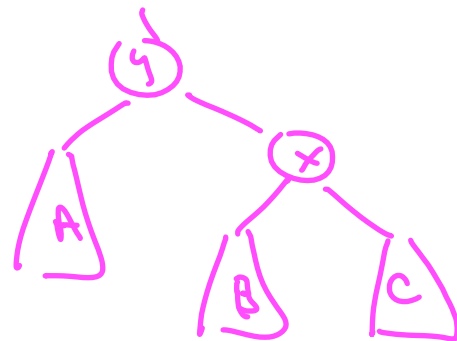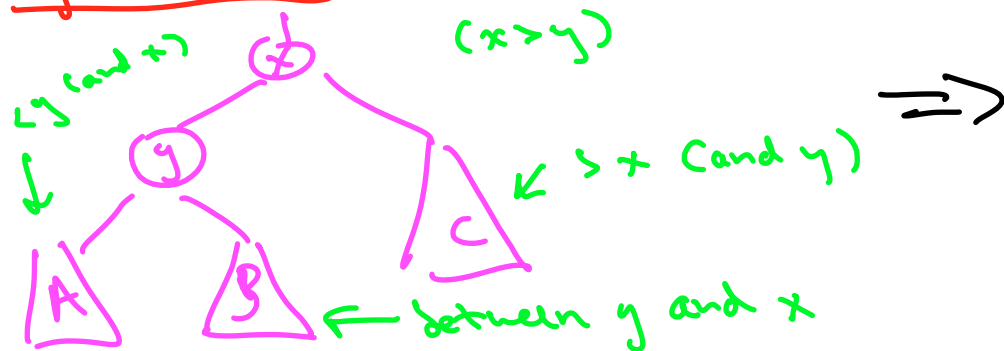
Idea: locally rebalance subtrees at a node in $O(1)$ time.

Left rotation: (of a parent x and right child y)

all rotations invoked on parent child pair
in BST. if right child of parent use left rotation.
if left child use right rotation

$(x<y)$

$\leq x$ (and y)

$> y$ (and x)

between x and y

$\Longrightarrow$

x<B<y

Tim Roughgarden

Right Rotations

Right rotation:

$(x > y)$

$> x$ (and y)

$< y$ (and x)

between y and x

Nice properties: Search tree property maintain, can implement in $O(1)$ time.