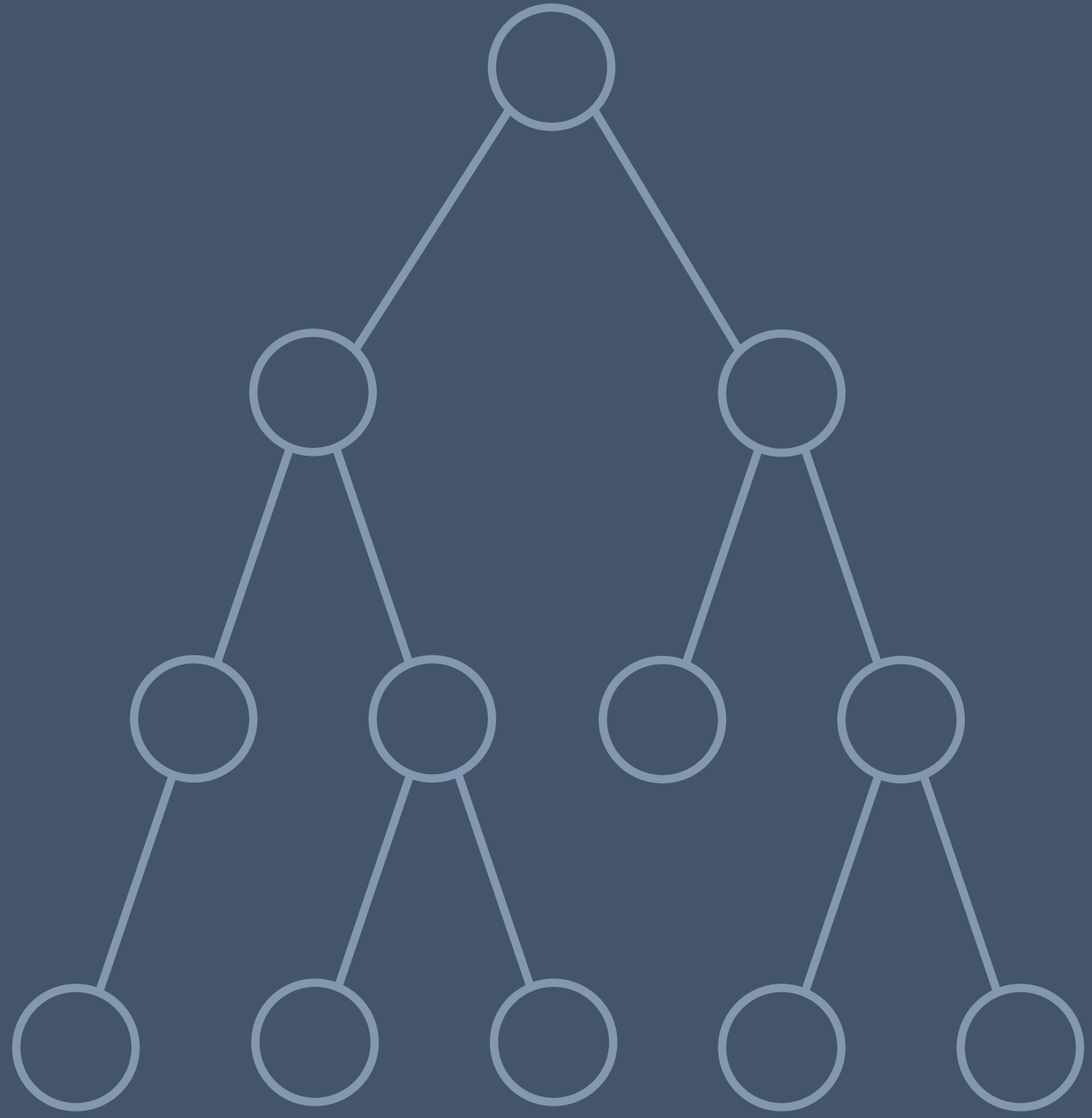
Binary search trees and K-dimensional trees

Trees and binary trees

A tree is a connected graph (no island) with no cycles.

We'll use rooted tree (only one node has no parent), specifically with 2 children (max) on each node.

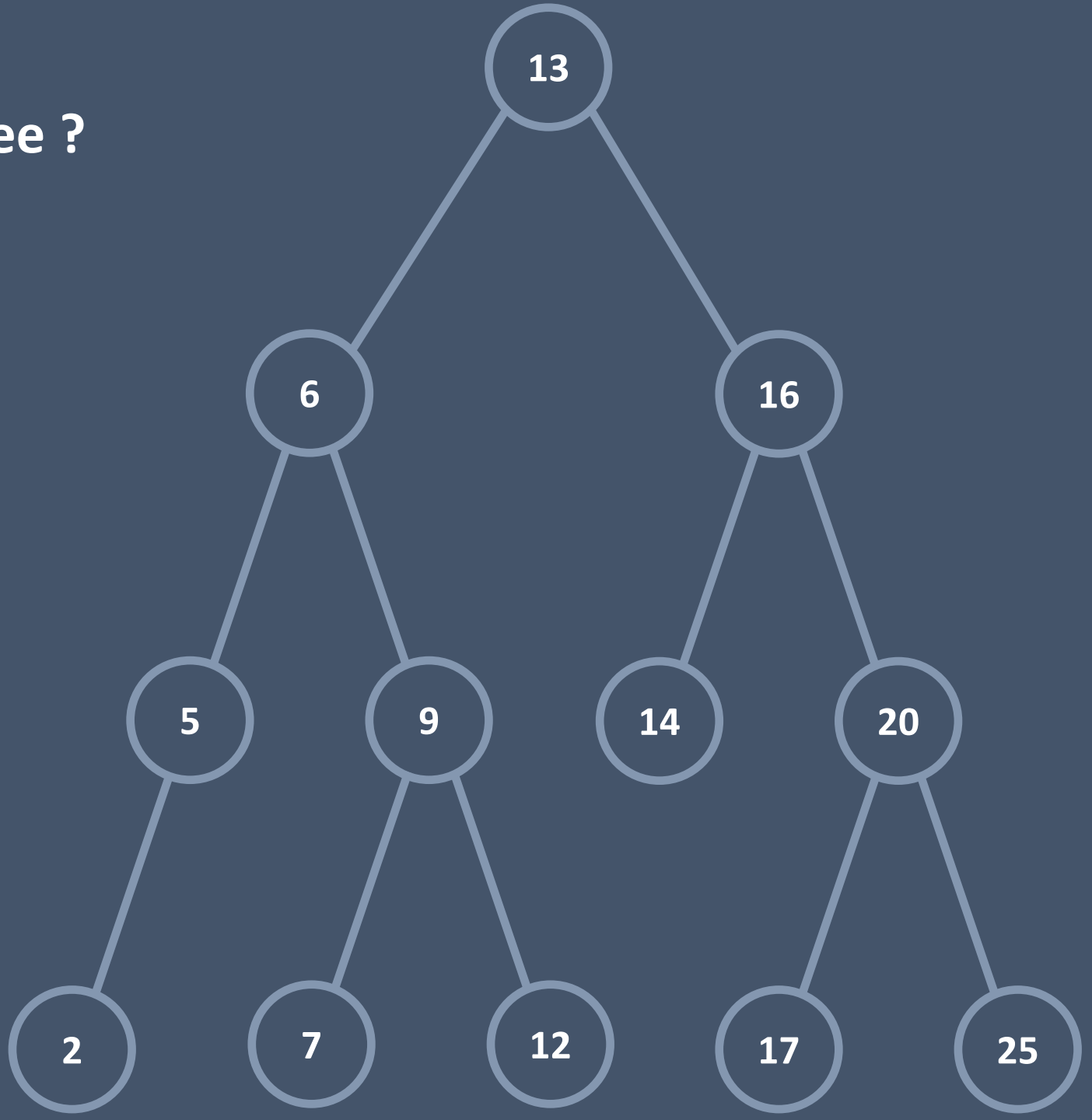


What is a binary search tree ?

There's only one rule : for each node
- all the nodes under its left child have lower values
- all the nodes under its right child have higher values

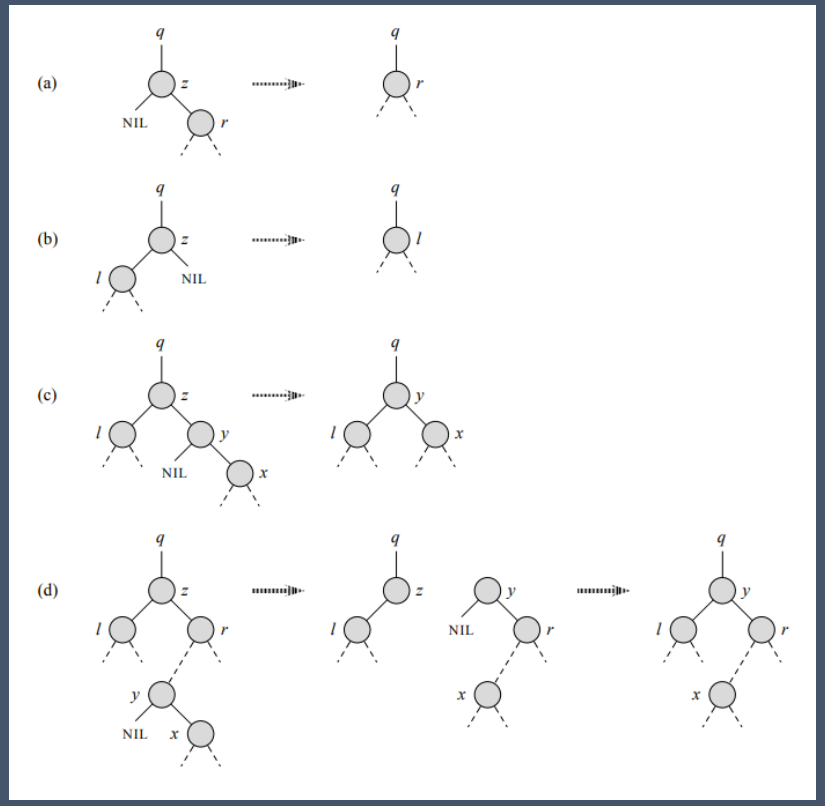
The only way to add a node is to add a leaf.

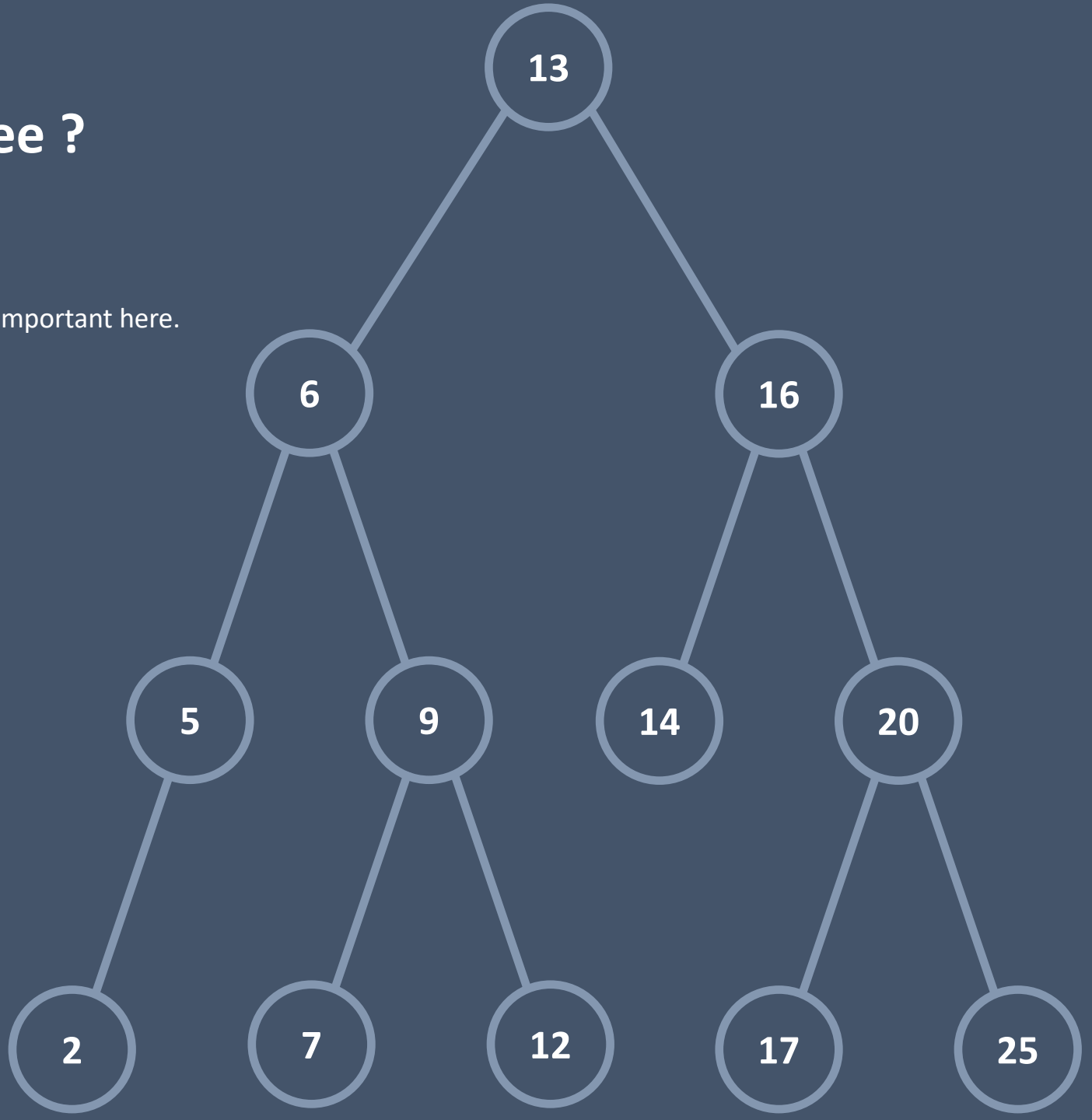
There are multiple types of binary search trees,
what differs between them are the methods for
adding and removing nodes.
For example, a balanced tree will optimize its structure
to have equal branches sizes to optimize search time.



What is a binary search tree ?

There are algorithms to remove nodes but it’s not really important here.

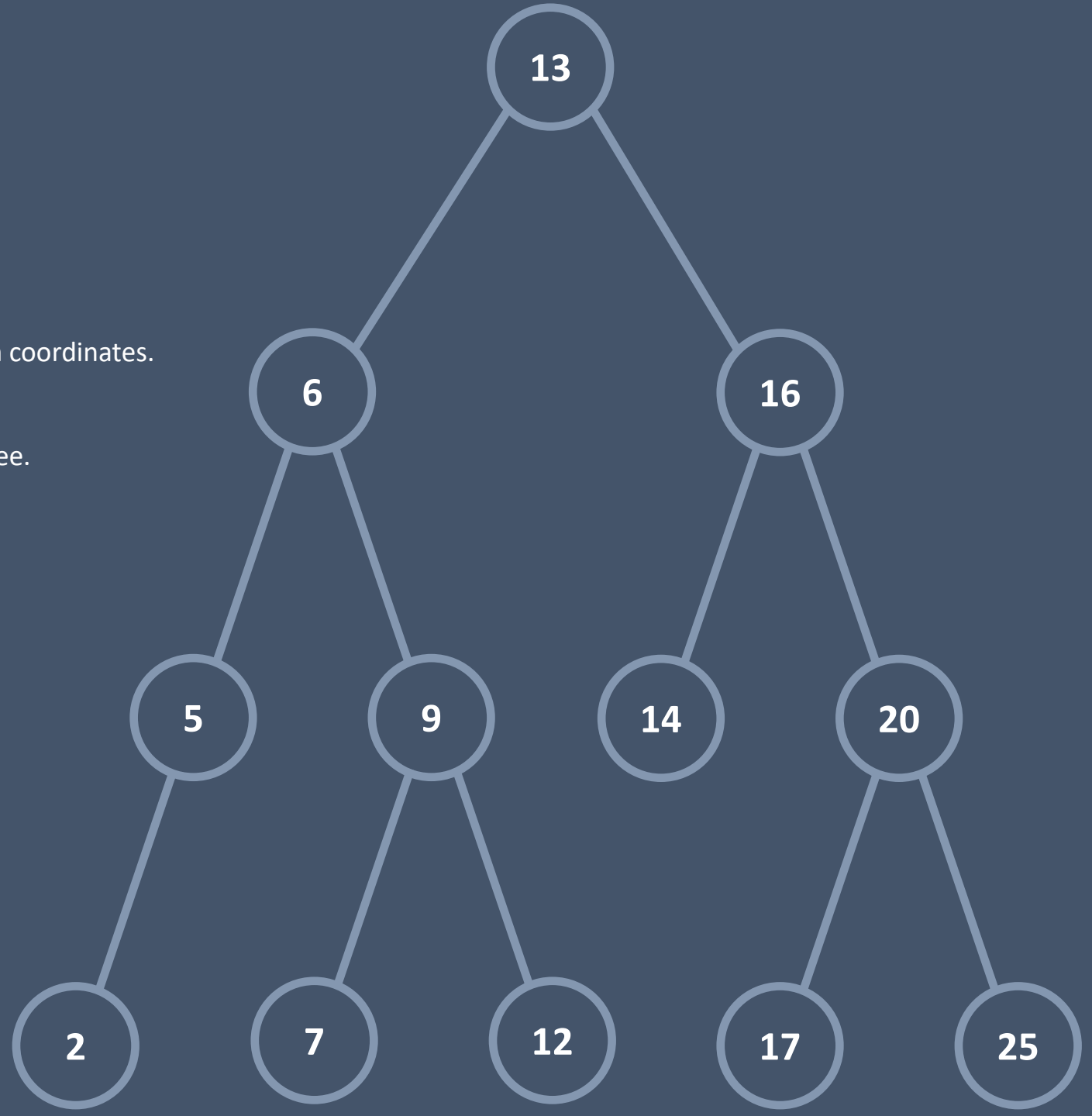




What it’s for ?

They’re used to search for points in logarithmic time,
the example I’ll use is searching for a point close to given coordinates.

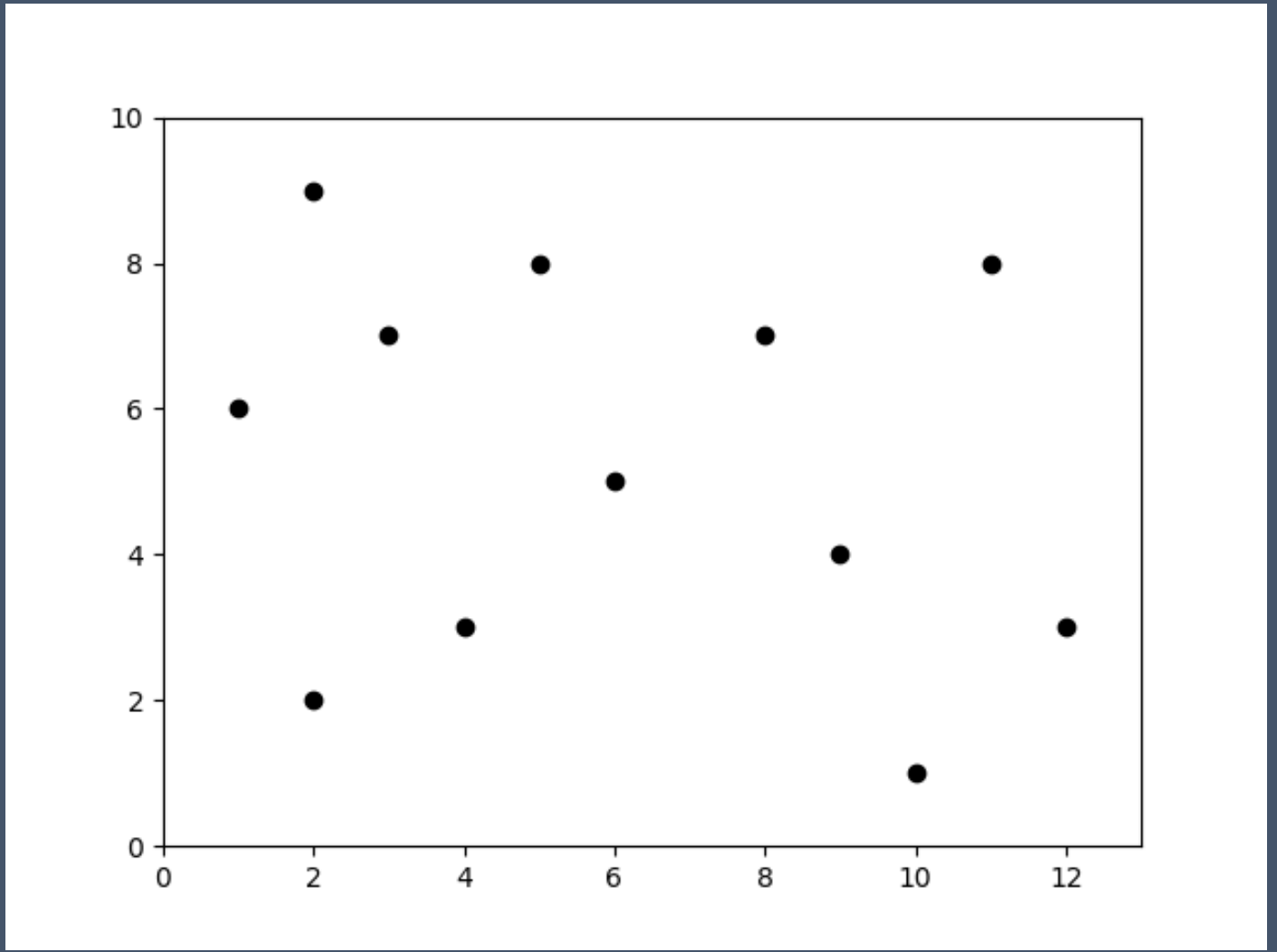
In 1D, the algorithm is pretty easy :
try to find the node with the value closer to 6.6 in this tree.



Multi dimensionnal BST : KD-trees

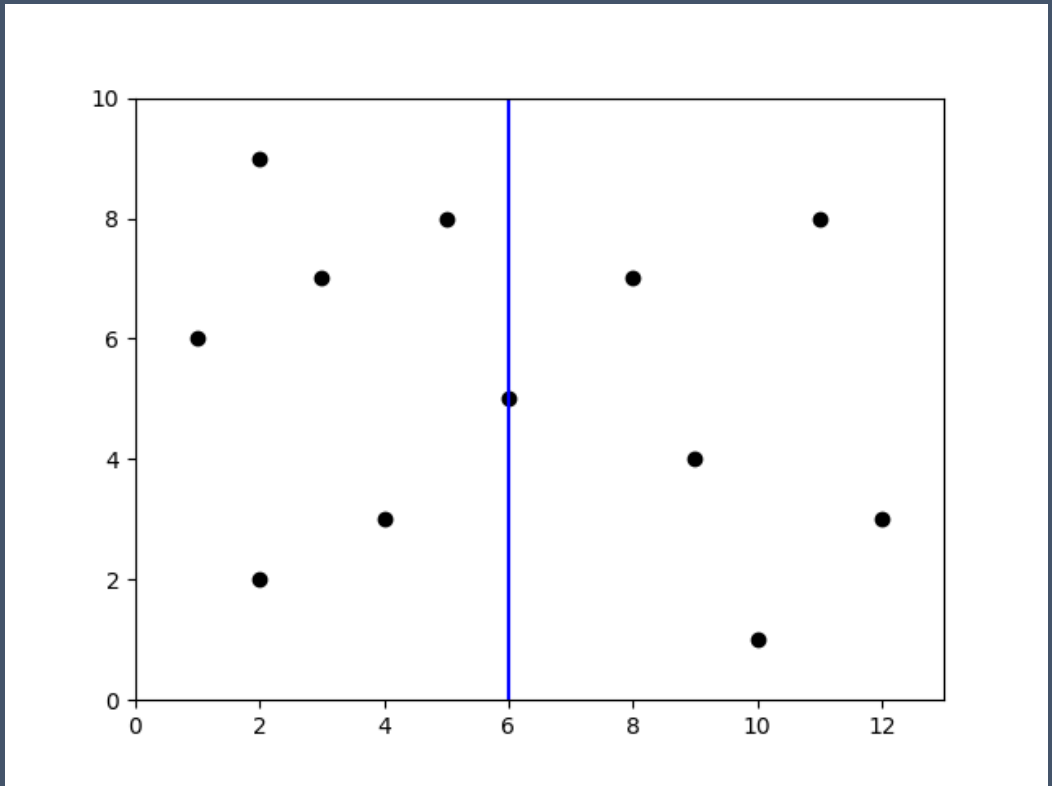
That’s where it gets complicated : here we’ll see 2D-trees but the search algorithm works in any dimension.

The tree is made so that each level divides a dimension. We’ll build the tree for this scatter :



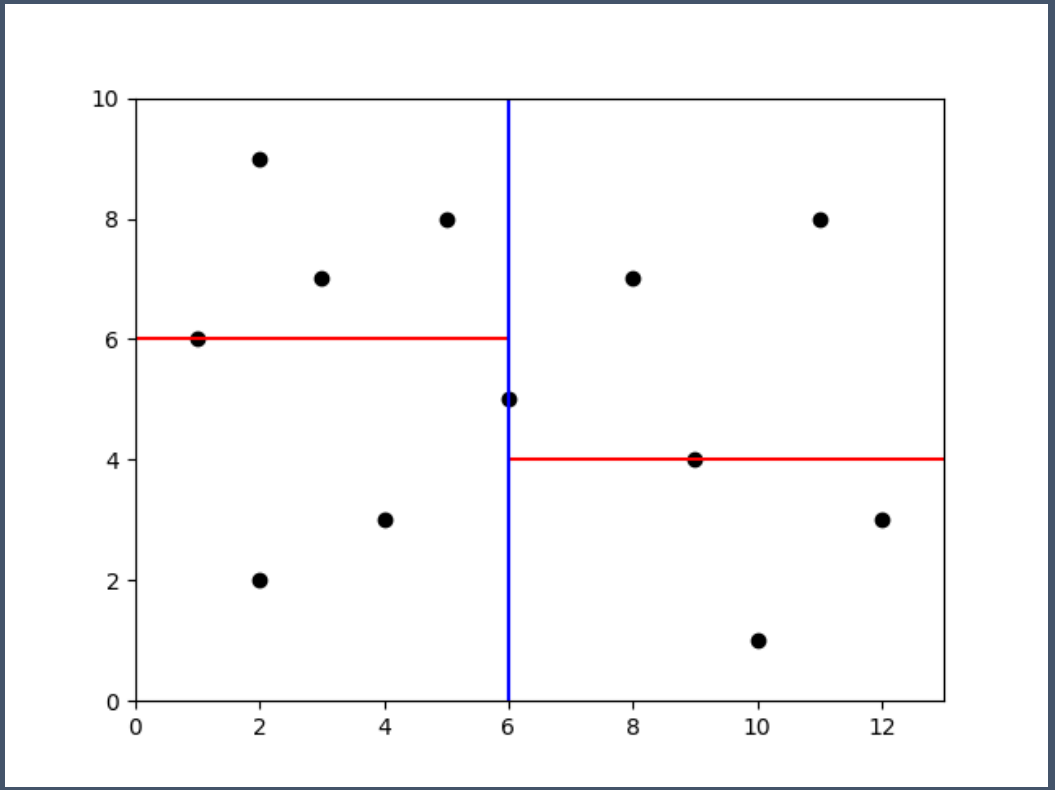
Multi dimensionnal BST : KD-trees

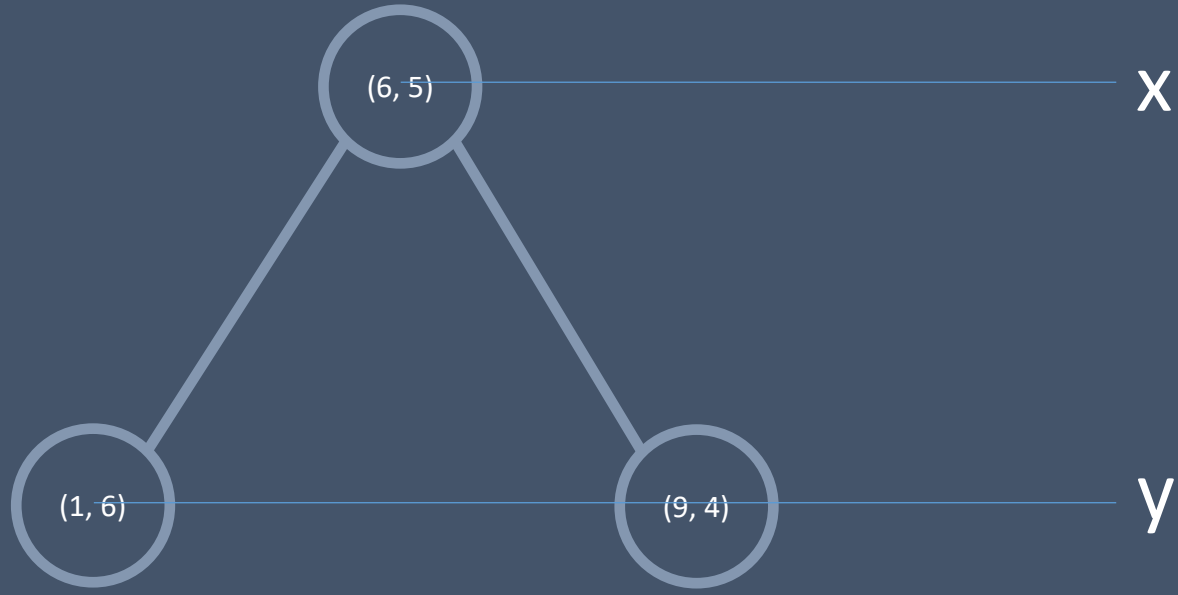
First, we cut the plan in half through the point in the middle of the x axis :



A 2D-tree

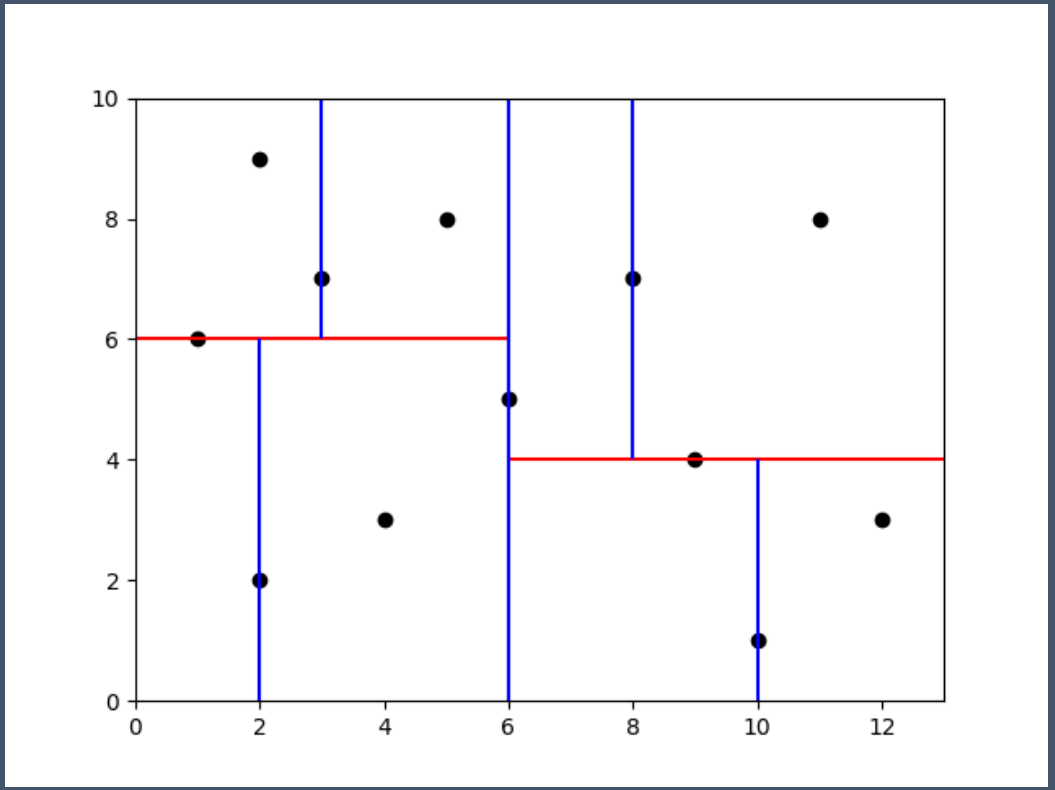
Then we cut the left part and the right part in halves :

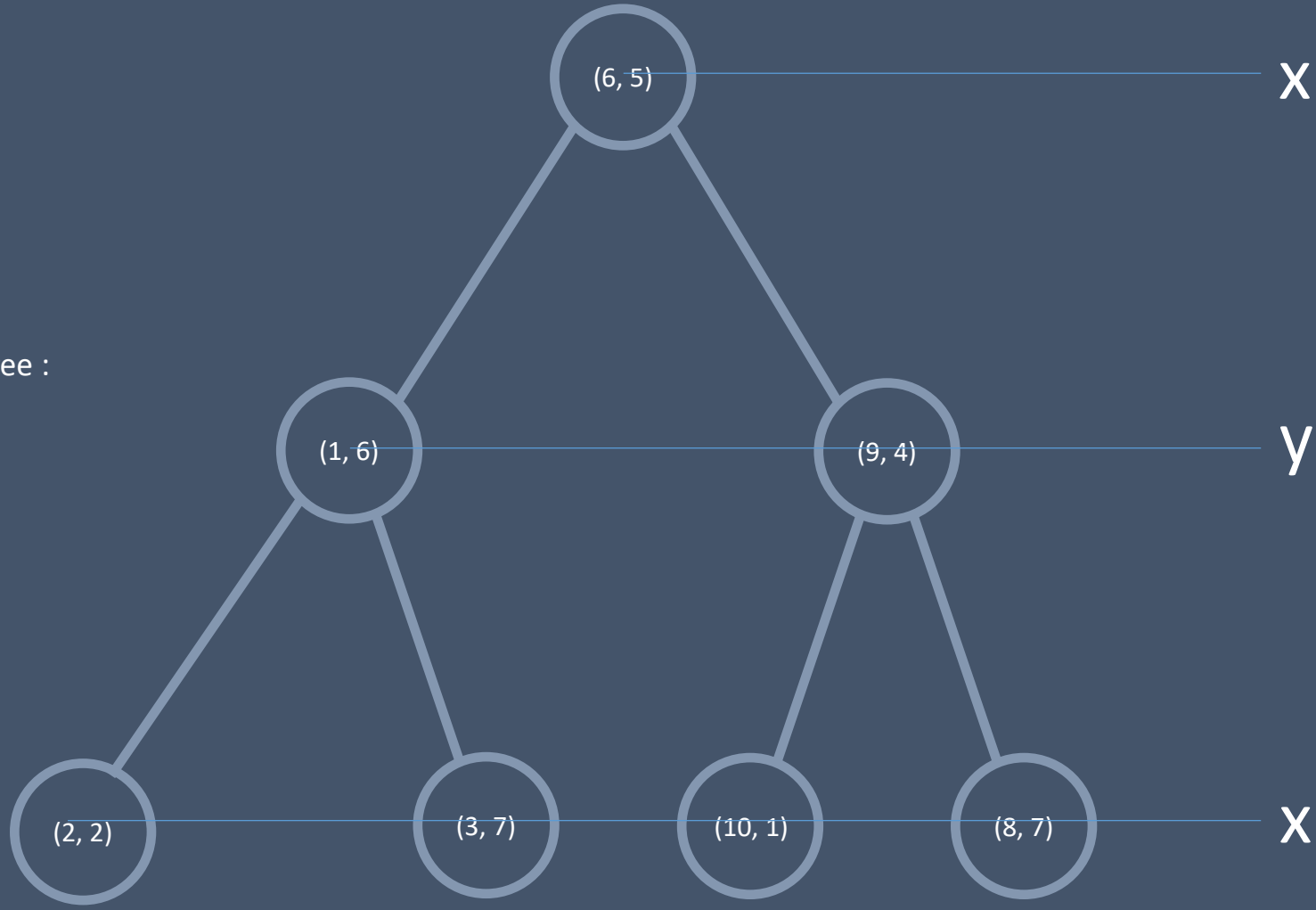




A 2D-tree

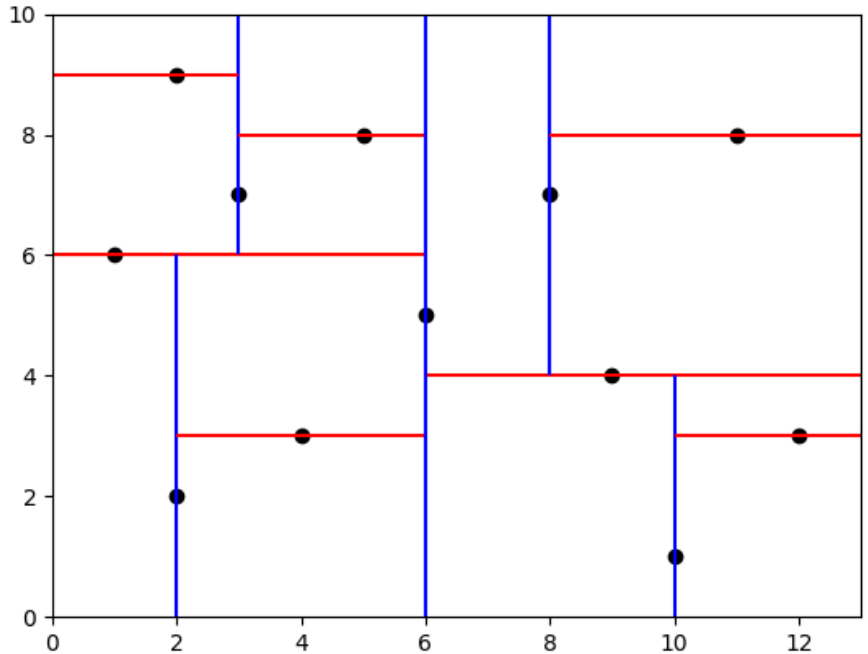
And then rince and repeat until all the points are in the tree :

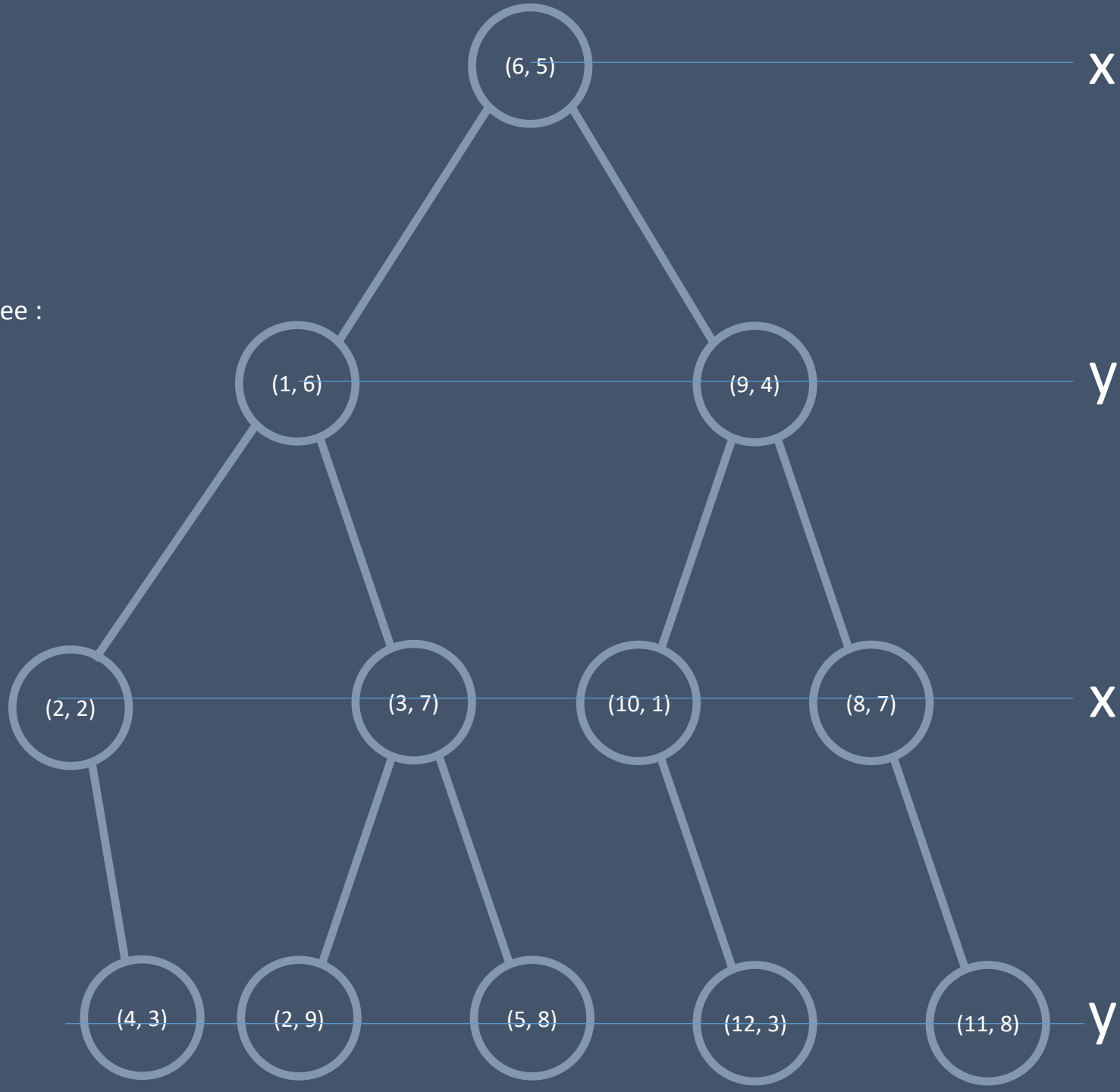




A 2D-tree

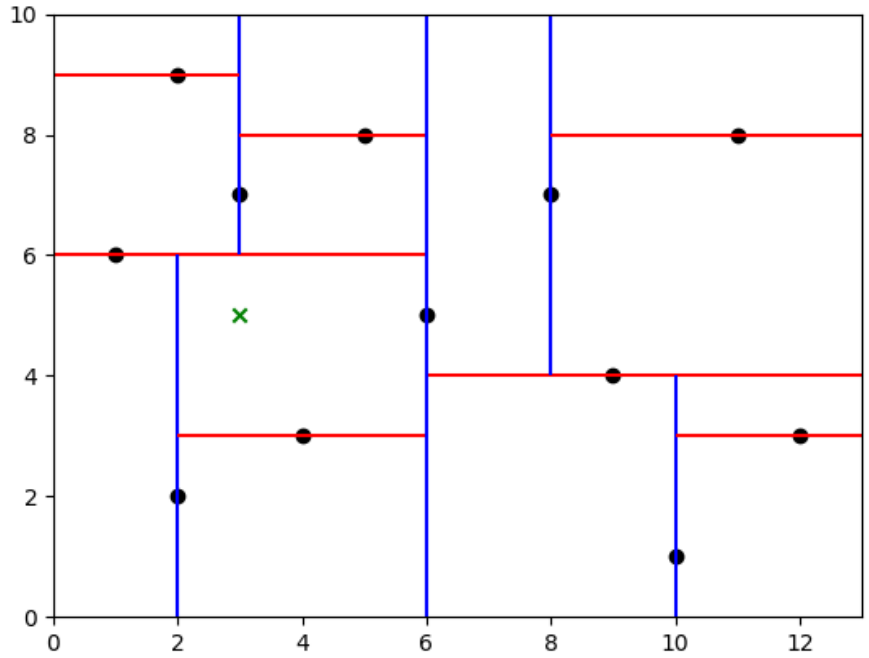
And then rince and repeat until all the points are in the tree :

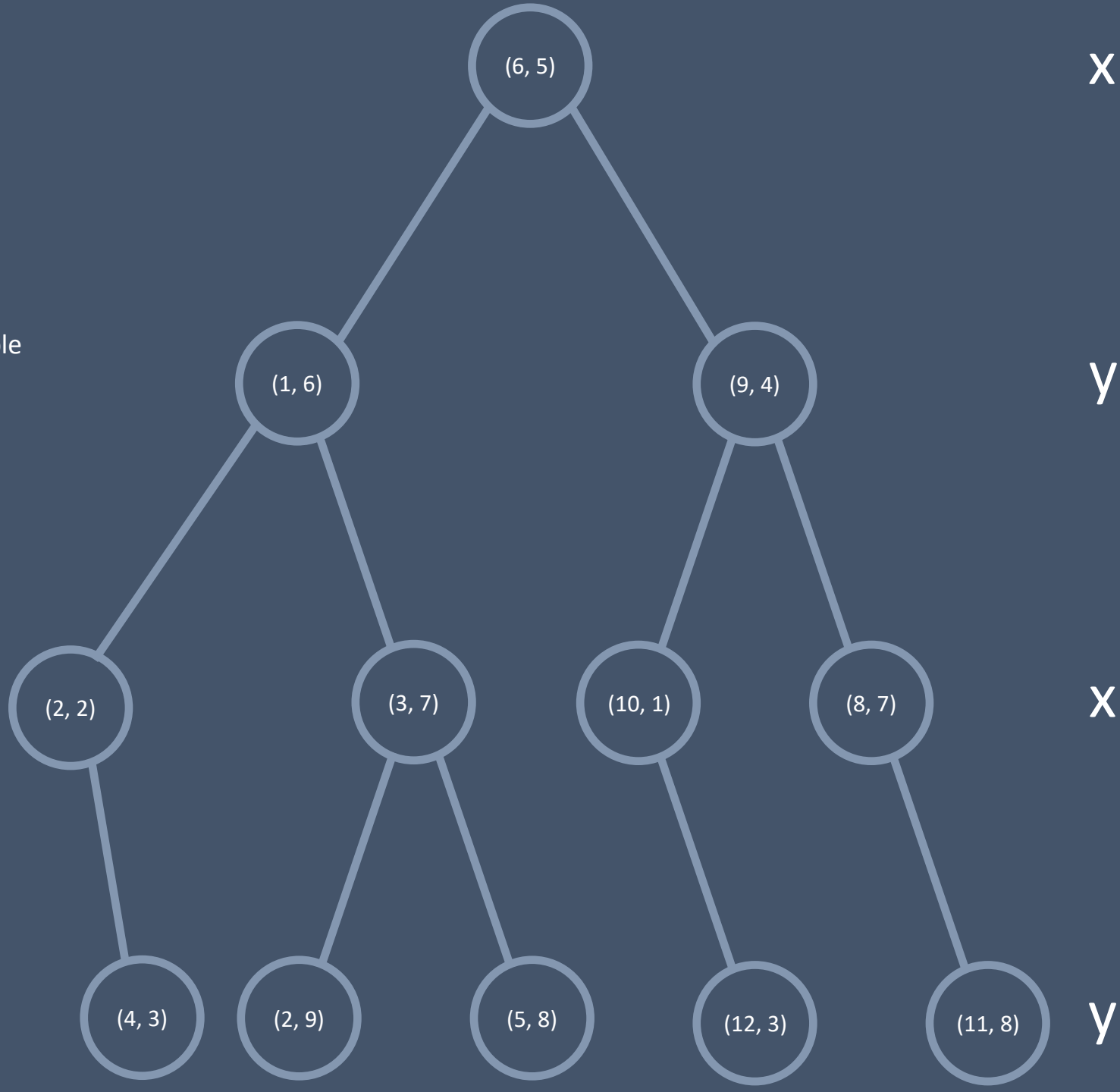




A 2D-tree

Now we can search which point of the scatter
is closer to given coordinates. Let's go through the example
of the point (3, 5) with euclidian norm.





exercises :

Easy :

https://www.hackerrank.com/challenges/binary-search-tree-insertion/problem?isFullScreen=true

Medium :

https://www.hackerrank.com/challenges/binary-search-tree-lowest-common-ancestor/problem?isFullScreen=true

Hard :

https://www.hackerrank.com/challenges/is-binary-search-tree/problem?isFullScreen=true