1 Proofs

1.1 BST Operations

Since AVL trees are based on BSTs, and the `lookup` and `bound` definitions can be used on AVL trees without any changes to the definitions, some proofs about these two operations were constructed.

```
lemma bound_false (k : nat) (t : btree α) :
  bound k t = ff → lookup k t = none := ...

lemma bound_lookup (t : btree α) (k : nat) :
  bound k t → ∃ (v : α), lookup k t = some v := ...
```

The two definitions were used together in these proofs, as they are linked to each other by virtue of their purpose. If a key is not bound in a tree, then lookup will not result in any node data being returned. If a key is bound in a tree, then some data will be returned.

The proofs with regards to `bound` and `lookup` and insertion is explained further.