Design Document

We choose to unpin almost instantly after we are done with using our read and alloc functions since we figured that by keeping pin utilization low we can keep our memory usage efficient. It also helps prevent us from unpinning an unpinned page as we unpinned it almost instantly.

We built helper functions to help our insertions be more readable and to to expedite our debugging process by isolating splits to its own function. We further split it into leaf and non leaf to be able to efficiently tailor the traversal algorithm. Specifically heavily using the right sibling relationship.

- We also added additional boolean flags to allow easy checks of the status our root nodes to make it easier to discriminate in certain functions such as startScan(). This allows us to have instant access to information such as leaf states. Allowing our algorithm to determine what route to take faster.

Pages that are left pinned are the root page and the header page because they are frequently accessed. And if the root page changes, the old root page is also remembered to unpin.

Duplicate keys would require an overhaul of our splitting process and require us to conduct additional tests on the condition of the tree.
Because say if we accidentally split keys between two separate nodes, it would complicate the searching process as we will need to look backwards. It would also create even further problems such as the key not pointing to the correct value once we start deleting as well.