Part a: Why the search cannot stop when a tombstone is encountered

When searching for a record in a hash table that uses open addressing and tombstones, we cannot stop the search at a tombstone. The reason lies in the insertion and deletion process:

1. **Insertion**: When an element XXX collides with a previously inserted element YYY, it will be placed further in the probe sequence.
2. **Deletion**: If YYY is deleted and the slot is marked as a tombstone, XXX remains in the table, but now lies beyond the tombstone.
3. **Search**: When searching for XXX, if we stop at the tombstone thinking the element is not present, we would erroneously conclude that XXX is absent.

Part b: Search for absent records with tombstone recycling

If we implement tombstone recycling during insertions, searches for absent records still need to proceed until an empty cell is found. This is because the tombstone slots might be occupied by records inserted after the tombstone was created.

Part c: Advantages of recycling tombstones

One significant advantage of recycling tombstones is that it reduces the average probe sequence length. When a new element is placed into a recycled tombstone slot, future searches for this element will encounter fewer probe steps, enhancing search efficiency.