Q1: How may you redesign the function prototype for identify, such that it more robustly returns the list of ad starts and ends?

To make this function more robust, we need to change the return value char\*. Since string value is error prone and hard to work with (more flexible in editing the data or deleting the data)

**Step 1: Define a struct to store each ad occurrence:**

typedef struct {

size\_t start;

size\_t end;

} AdOccurrence;

**Step 2: Define a result struct to wrap occurrences and their count:**

typedef struct {

AdOccurrence\* occurrences; // Dynamically allocated array of results

size\_t count; // Number of matches found

} IdentifyResult;

**Step 3: Update the function signature:**

IdentifyResult tr\_identifstruct sound\_seg\* target, struct sound\_seg\* ad);

Now we don’t need to worry about string-based return format ("<start>,<end>\n..."). Also, you can test it easily since we have a count variable in the struct. As well as, tracking memory easier because returning a struct.

Q2: Referring to REQ 1.1, how did you identify which track is responsible for which memory, and how did you ensure that all memory is freed? If you were not successful in ensuring, how did you plan to?

For Linked‐List Nodes, each node only points to one track, nodes are allocated and attached to that track’s head pointer. Therefore, for linked list nodes, we always know which track in which memory. So, we just look for the head pointer in that linked list and we will know where it is in memory.

However, for SampleData, each SampleData has a ref\_count field indicating how many different tracks refer to it.

To ensure that all memory is freed we need to free all the nodes just as what my code did. (Go to track’s head node and while there is curr node, I free it. Until all nodes are freed then I free the track).

Plan for freeing SampleData.

* Store a ref\_count in each SampleData.
* Increment ref\_count each time a track’s node points to that sample
* In tr\_destroy, decrement ref\_count for each node’s SampleData
* If ref\_count == 0, call free on the SampleData\_ptr