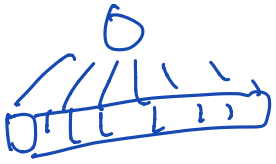
Asgn 7 Design

Workflow

1. Work on tie.c
2. Work on word.c
3. Work on io.c
4. TBD.

Trie.c

① Each Node has a children array, size of 256 ASCII.



the index is the array could be empty.

② Every time we try to put something in the Array.

Ex: Let's say we want to pass in "a"

uint8_t value = 97.

So root → children[97] = code

All other index will remain empty.

Pseudocode.

Trie.h

trie_node_create:

malloc (trie_node) // allocate memory for node

malloc (trie_node -> children) // allocate memory for array.

trie_node_delete:

free (node) // free this node only
// array remains still

trie_create:

malloc (root) // first node in the tree, has code of EMPTY code.

trie_reset

for all children:
delete (children) // delete all children except the node itself.

trie_delete

for all children
delete (children) // delete children

free (root) // also delete this root.

trie_step:

return n -> children[sym]

// return the portion of this sym.

/

Word.c

word_create: (arr: syms, len)

w -> syms = syms // syms is a long array

w -> len // we need len to tell us how many nodes of this array is actually used

word_append

w -> syms[len] = sym // modify the last index

len++ // change len so we know we modify the last node.

word_delete:

free(w -> syms) // delete the array

free(w) // delete this object.

wt- create :

malloc (wt) // allocate memory for object

malloc (wt-> array) // word allocate memory for the array

// size of Max - CODE

wt > array [0] = " " // initialize array

return wt

wt - reset :

for i to word table . size

wt. delete // delete every thing beside array [0]

wt. delete :

for i in word table

word . delete (i) // free all word in table

free wt // free object