15122 Bootcamp

Debugging Data Structures

Spring 2024

tinyurl.com/122dbug2M

Topics

01

Printing structs (5 mins)

02

Data structure invariants (25 mins)

03

Test cases (20+ mins)

tinyurl.com/122dbug2M

01

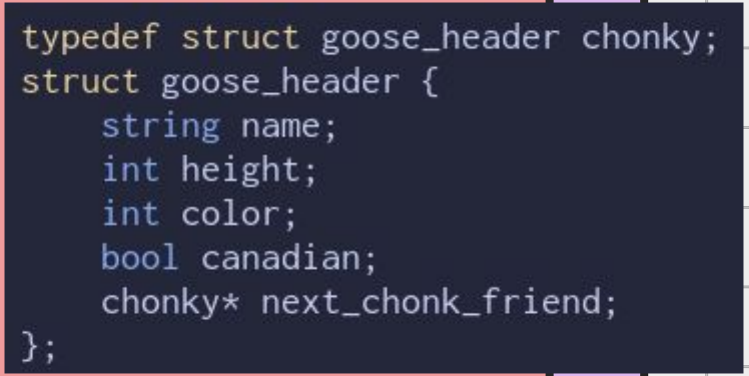
Printing Structs

tinyurl.com/122dbug2M

Print Data Structures

Scenario: *we have a Goose structure with*

- **Name** (string)
- **Height** (int)
- **Color** (int - categorical)
- **Canadian-ness** (bool)
- **Friends** (linked-list)



```c
typedef struct goose_header chonky;
struct goose_header {
    string name;
    int height;
    int color;
    bool canadian;
    chonky* next_chonk_friend;
};
```

Goose Printer

```
void printList(chonky *node) {
    printf("\t\tFriends: ");
    while (node != NULL) {
        printf(" %s - ", node->name);
        node = node->next_chonk_friend;
    }
    printf("\n");
}
```

```
void printGoose(chonky *honk) {
    //name
    printf("\tName: %s\n", honk->name);
    //integer
    printf("\t\tHeight: %d inches\n", honk->height);
    //category
    printf("\t\tColor: ");
    if (honk->color == 1)       printf("black\n");
    else if (honk->color == 2)  printf("orange\n");
    else if (honk->color == 3)  printf("white\n");
    else                        printf("no color\n");
    //boolean
    string from = honk->canadian ? "\t\tFrom: Canada\n"
                : "\t\tFrom: not Canada\n";
    print(from);
    //linked list
    printList(honk->next_chonk_friend);
}
```

Output

```
    Name: Kevin
            Height: 13 inches
            Color: white
            From: Canada
            Friends:  Allen -  Jeffrey -  Alex -
```

02

Data Structure Invariants

Jessica

Questions to ask yourself

NULL?

Can it be NULL?

Structure?

Is my data structure intact?

(e.g. is_segment)

Correct?

Is the content of my data structure correct?

Field Invariants?

Does my data structure follow the rules of the question?

NULL?

If passed a pointer… is it NULL? Can it be NULL?

Should return true/false… **NOT** a contract

(otherwise NULL checks won't happen outside of -d mode)

Structure?

Does data structure have a specific format, ie how do multiple structs connect and interact?

How does this vary? (dictionaries, linked lists, etc)

Examples:
- is_segment
- is_circular
- is_balanced (in the future)

Correct?

Are the values correct? What can I check?

Examples

- Linked list in sorted order (call is_sorted)
- within-struct pointers are NULL/not NULL
- Values are correct based on specifications
    - (e.g. pixel between 0 and 255)

Rules?

Are there rules about the length of items?

We **MIGHT** use contracts here because we are forced to in c0/c1

- \hastag
- \length

Your Turn!

Task 1: Writing *is_person* (10 mins)

- name	(their name)
- left	(holding hands on left)
- right	(holding hands on right)
- best_friend	(their best friend)
- age	(their age)
- shirt	(their shirt color)
- in_friend_group	(if they are in group)

Check the exercise doc for rules!

Additional challenge exercise: shirt colors requirement

tinyurl.com/122dbug2M

Work with your neighbors!

is_person

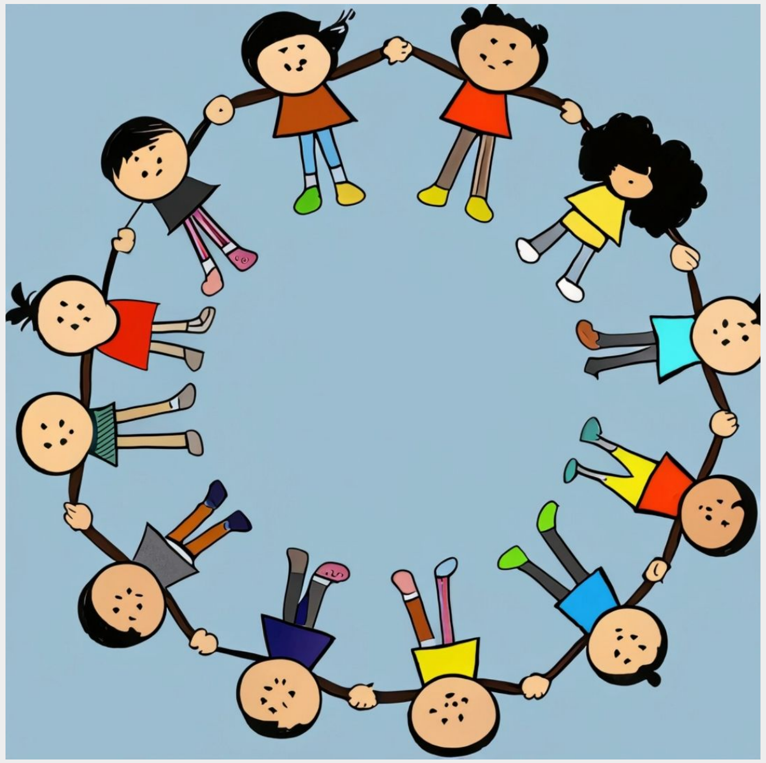
Your Turn (cont.)

<u>Task 2:</u> ***is_friend_group*** **helpers (10 mins)**

Finish a helper called "age_increasing" to check the increasing age requirement of a friend group and implement the unique shirt requirement for is_person if you haven't already!

Check the exercise doc for more details!

Additional challenge exercise: write the mutual_friends *helper*



Work with your neighbors!

is_friend_group

03

Test Cases

How to come up with Test Cases

Incorrect structures
Test *your* data structure invariants

Incorrect content
Test the contents of your data structure and invariants

Edge cases

You have heard this before

Stress testing

Small loops going through range of values

Common Arbitrary Cases

Basic cases

- (like the writeups)

A quick reference

	Test passes	Test fails
When it should	yay!	yay!
When it shouldn't	Invariants/contracts too weak	Invariants/contracts too strong or incorrect

Tips and Tricks

Test cases are for **YOU**

- write tests before and as you go
- use them to figure out where bugs originate

Compartmentalize

- Write test cases to test **one** issue

Be **efficient**

- write helpers to test repetitive stuff
- write helpers to **make** data structures
- **use** your specification functions
- decompose the data structure (types)

Your turn! Test cases

Task 3 & 4: Testing *is_person* and *is_friend_group* (20+ mins)

Check the exercise doc for details!

Most of the helpers are written for you as an example.

You will have to write *make_person* and *make_friend_group*.

Then, write test cases to test your own implementation and *our buggy* implementations of is_friend_group.

is_friend_group

THANKS!

https://tinyurl.com/122dbug2