

## Outlook:

1	Notes on using malloc/calloc
2	Dynamic Arrays
3	Implement Stacks using arrays [ exercise 8]
4	Understanding linked lists & supplied program
5	Exercise 9
6	Other small exer on linked list & array of linked lists
	Ass 2: understanding

# Review on: malloc/calloc/realloc

```
XXX = malloc( ? * sizeof( *XXX) ); OR  
XXX = calloc ( ? , sizeof(*XXX) );  
assert (XXX);
```

```
XXX = realloc(XXX, sizeof(XXX));  
assert(XXX);
```

```
free(XXX);
```

- Always use assert right after malloc/calloc/realloc
- one free for each malloc /calloc (not for realloc)
- With declaration `int **p :`
  - `*p` invalid before `p= calloc(?, sizeof(*p));`
  - `**p` invalid before `*p= calloc(?, sizeof(**p));`

# Arrays: Dynamic Memory Allocation

Need an array of `data_t`, but with a problem:

**the array's size has no upper bound.**

How to declare and manage this array?

## Exercise 8

Stacks can also be implemented using an array of type `data_t`, and static variables. Give functions for `make_empty_stack()` and `push()` and `pop()` in this representation.

# 5 minutes break (from programming)

# Quiz: What's your tutor's name?

- A. Alistair "Algorithms-Are-Fun"
- B. Anh Vo
- C. Artem Polyvyanyy
- D. Alistair Moffat

and

Please do Tutor Quality of Casual Teaching (QoCT) survey (right now!) for COMP10002 by:

Goto Link:

<https://apps.eng.unimelb.edu.au/casmas/index.php?r=qoct/feedback&subjCode=COMP10002>

OR visit the first link from googling:  
[unimelb tutor survey comp10002](#)

## Exercise 9

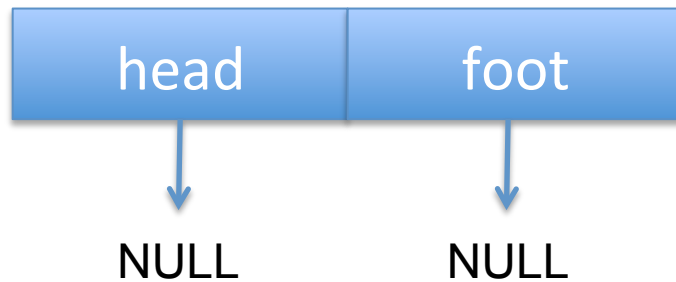
Suppose that insertions and extractions are required at both head and foot. How can `delete_foot()` be implemented efficiently? (Hint, can a second pointer be added to each node?)

OR some other small exercises?

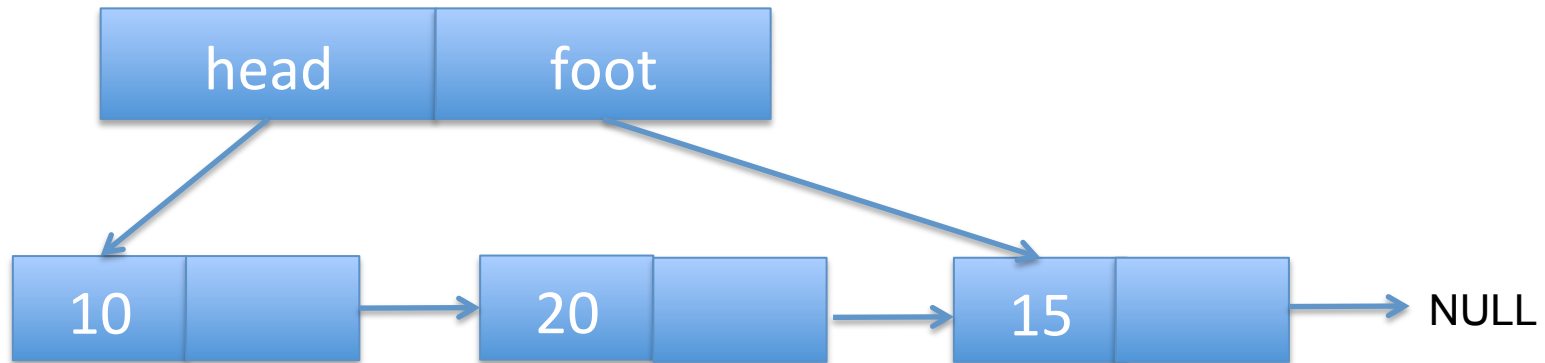


# Alistair “Algorithms Are Fun” Moffat’s lists in listops.c

Empty list:



Non-empty list



# Assignment 2: The Programming Task – Q&A