You’re the jury – whose fault was this

COMP1511/COMP1911 Week 4!

M13B: 1pm – 4pm || T11X: 11am – 2pm

Tutors: William (me!) + Jason || Daniel

My GitHub:

https://github.com/william-o-s/unsw_comp1511_tutoring

Please remember:

Census Date

Reminder: This Sunday is Census Date, please chat to your tutor if you have any questions about it.

Revision Sessions

We will be running some revision sessions this week (Week 4) to help you catch up and consolidate course content we have covered so far! Revision sessions are flexible in its structure aimed to cater for the attendees of the session. Please sign up for the revision sessions via this link (Access code: "COMP1511").

The sessions will run:

- Week 4 Wednesday 06/03 11am-1pm - Bass Lab OMB149B
- Week 4 Thursday 07/03 10am-12pm - Online via Help Session Microsoft Teams

Assignment 1

Reminder: Assignment 1 has been released, is avalible on the course website here and is due Monday, 8 July 20:00. It is a recommendation that you start early and watch the Assignment 1 Livestream happening on Week 4 Wednesday, March 14:30-15:30 online via the youtube link.

Tutorial Agenda:

Part 1

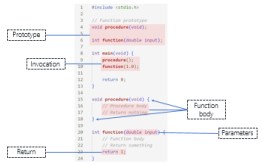
Part 2

Part 3

What are these? Have you heard these names before?

```
int array[3] = { 1, 2, 3 };
double list[3] = { 1.0, 2.0, 3.0 };
char collection[3] = { 'a', 'b', 'c' };
```

Let's look at some characteristics of functions in C



Applications of functions

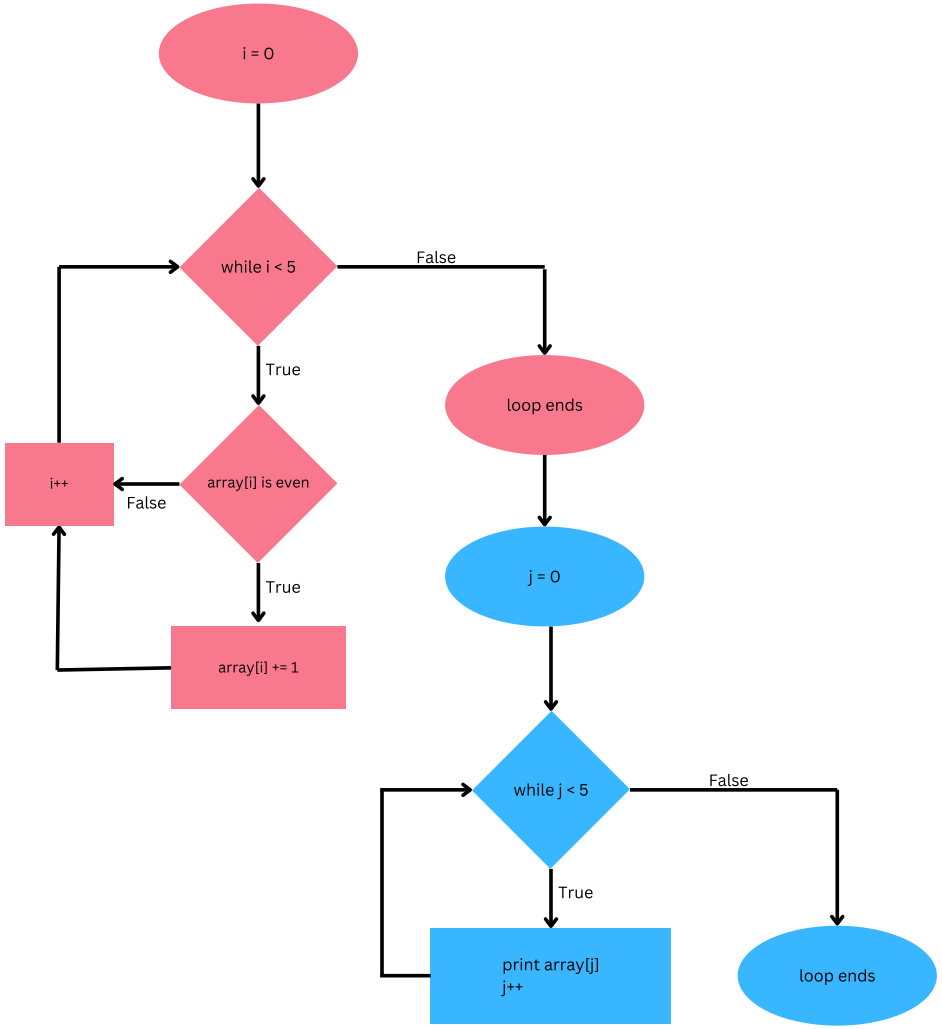
- Printing out values from an array
- Checking inputs from the user are valid (e.g. within a valid range of values)
- Modifying arrays in a specific way (e.g. sorting an array of ints in ascending order)
- Searching for a particular value in a collection such as an array
- Mathematical operations which require multiple lines of code
- Handling and printing error messages
- Memory allocation and value initialisation for a data structure (this will be addressed later in the term)

generally, any repetition or abstraction

What are these? Have you heard these names before?

```
int array[3] = { 1, 2, 3 };
double list[3] = { 1.0, 2.0, 3.0 };
char collection[3] = { 'a', 'b', 'c' };
```

Let’s debug the starter code to match this flowchart



Debugging Techniques

- Using printf
- Reading dcc output
- Using dcc-help
- Using dcc-sidekick

Your turn! Edit the starter code for these tasks

Your turn! (10 mins)

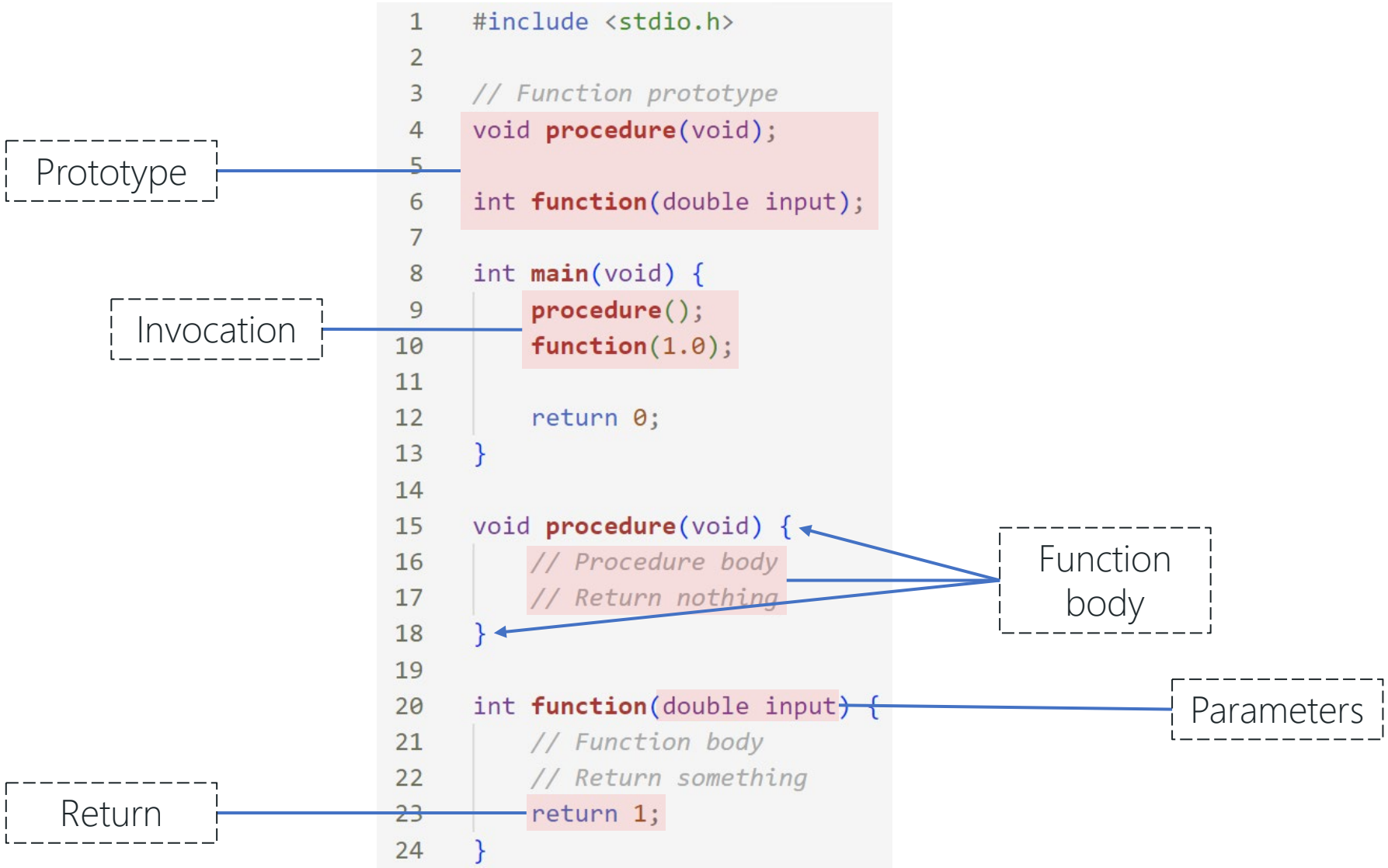
Copy Array

- Create an array of doubles with 3 elements, each with a non-zero value.
- Create another array of doubles with 10 elements where every element initialised to `0.0` .
- Create a while loop that loops through every element of the first array.
- Copy the elements of the first array into the second array (leave 0's at the end)
- Create a while loop that prints out all the elements of the second array.

Largest Character

- Create a character array with exactly 8 elements.
- Create a character variable called `largest_character` , equal to the first character of the array.
- Create a while loop to loop through the character array.
- Create an if statement to check if the current character has a higher ascii value than "largest_character"
- Print out the largest character you've found.

Let's look at some characteristics of functions in C




```c
#include <stdio.h>

// Function prototype
void procedure(void);

int function(double input);

int main(void) {
    procedure();
    function(1.0);

    return 0;
}

void procedure(void) {
    // Procedure body
    // Return nothing
}

int function(double input) {
    // Function body
    // Return something
    return 1;
}
```

Let’s run through a demo

Your turn! Edit the starter code for this

Your turn! (20 mins)

We have been provided a program which simulates a bubble tea shop and is very similar to last weeks coffee shop activity. All the code for the program is currently in the `main` function and this task involved you working together to refactor the program to use a series of functions.

```
// bubble_tea.c
//
// Written by YOUR-NAME (YOUR-ZID) on TODAYS-DATE
//
// This program is a simple bubble tea store used to teach functions
```

Need a hint? Define these function prototypes

```
/* return type */ check_stock(/* parameters */);
/* return type */ calculate_cost(/* parameters */);
/* return type */ update_stock(/* parameters */);
/* return type */ print_order(/* parameters */);
/* return type */ print_inventory(/* parameters */);
```

Applications of functions

- Printing out values from an array
- Checking inputs from the user are valid (e.g. within a valid range of values)
- Modifying arrays in a specific way (e.g. sorting an array of ints in ascending order)
- Searching for a particular value in a collection such as an array
- Mathematical operations which require multiple lines of code
- Handling and printing error messages
- Memory allocation and value initialisation for a data structure (this will be addressed later in the term)

generally, any repetition or abstraction

If time permits, let's Kahoot! (otherwise pack up)