

COMP1511 PROGRAMMING FUNDAMENTALS

LECTURE 17

Starting Revision

LAST WEEK...

- Multi-files
- More linked lists

FOR OUR FINAL WEEK...

- REVISION!

“

WHERE IS THE CODE?



Live lecture code can be found here:

[HTTPS://CGI.CSE.UNSW.EDU.AU/~CS1511/22T3/LIVE/WEEK10/](https://cgi.cse.unsw.edu.au/~cs1511/22T3/LIVE/WEEK10/)

“

COURSE FEEDBACK



Tell us about your experience and shape the future of education at UNSW.

Click the link in Moodle

Please be mindful of the [UNSW Student Code of Conduct](#) as you provide feedback. At UNSW we aim to provide a respectful community and ask you to be careful to avoid any language that is sexist, racist or likely to be hurtful. You should feel confident that you can provide both positive and negative feedback but please be considerate in how you communicate.



my Experience surveys
<http://myexperience.unsw.edu.au/>

ARRAY OF STRUCTS

- An array of structs is really just an array, where each index of the array indexes a struct, so for example:

```
struct direct {  
    int number;  
    char dir;  
};
```

```
struct direct array[4];
```

- Creates an array, where each element of the array has a struct direction in it (and there are four in total!)

```
struct direct  
{  
    int number;  
    char dir;  
};
```

0

```
struct direct  
{  
    int number;  
    char dir;  
};
```

1

```
struct direct  
{  
    int number;  
    char dir;  
};
```

2

```
struct direct  
{  
    int number;  
    char dir;  
};
```

3

ARRAY OF STRUCTS

- To access an array of struct, we use the same syntax, we refer to the name of the array and the index at which we want access. And then we access whichever member we want with a .
- So if I want to access the second index of the array and assign something to the dir member

```
struct direct array[4];
```

```
array[2].dir = 'a';
```

```
struct direct  
{  
    int number;  
    char dir;  
};
```

0

```
struct direct  
{  
    int number;  
    char dir;  
};
```

1

```
struct direct  
{  
    int number;  
    char dir;  
};
```

2

```
struct direct  
{  
    int number;  
    char dir;  
};
```

3

ARRAY OF STRUCTS

- Loop through an array of structs and gather some kind of information

Given an array of structs, where each struct is:

```
struct direction {  
    int number;  
    char dir;  
};
```

Print out the total of the number of steps taken in a specific direction. So for example, if direction is 'l', find all the structs with direction as 'l' and add the numbers in those structs up. Edit the function

```
int total (int size, struct direction array[MAX])
```

total.c

ARRAY OF STRUCTS

Given an array of structs, where each struct is:

```
struct animal {  
    int number;  
    char animal;  
};
```

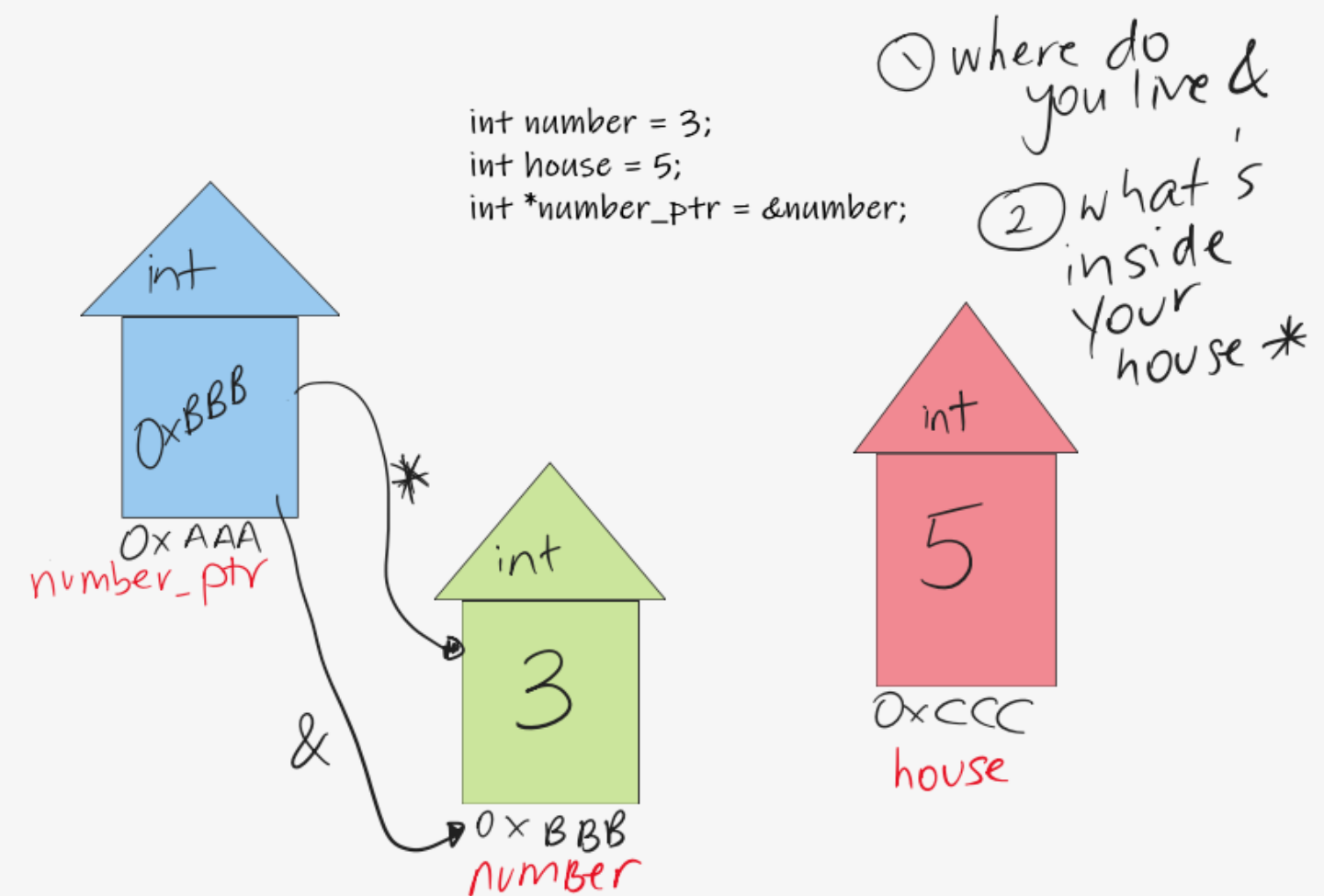
Print out the average number of dogs in each family ('d').

```
int average (int size, struct animal array[MAX])
```

REVISION TIME!

POINTERS

- Pointers are another variable type in C
- Pointers store the memory address of another variable
 - & - gives the address of
 - * - dereferences a pointer, so provides the value of stored at the address the pointer is at

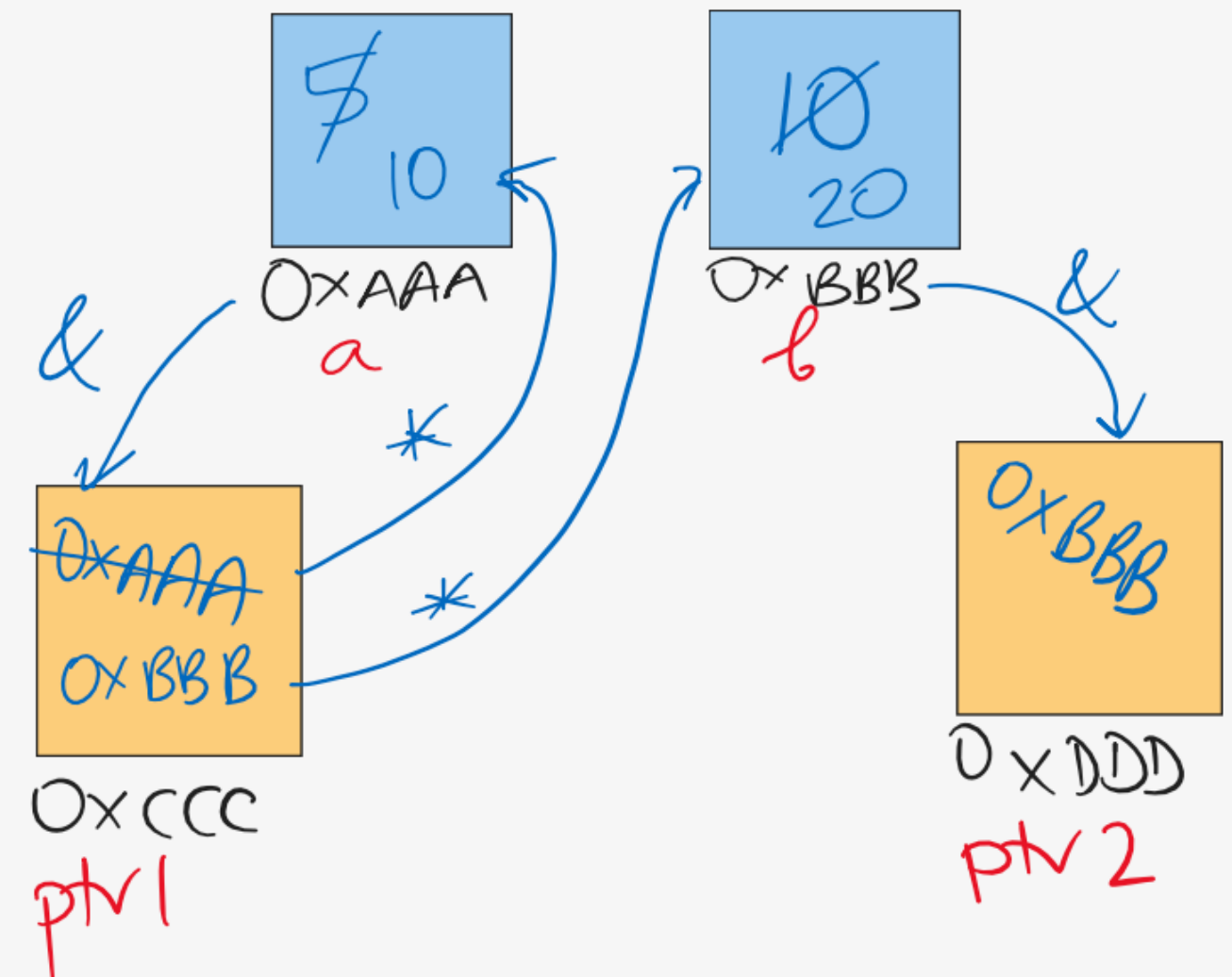


REVISION TIME!

POINTERS

- Let's see an example:
`pointer.c`

```
int main(void) {  
    int a = 5;  
    int b = 10;  
    int *ptr1;  
    int *ptr2;  
    ptr1 = &a;  
    ptr2 = &b;  
    *ptr1 = 10;  
    ptr1 = ptr2;  
    *ptr1 = 20;  
    printf("a = %d\nb = %d\n", a, b);  
    return 0;  
}
```



REVISION TIME!

YOUR TURN FOR POINTERS

- Write some programs using pointers to:
 - Swap two numbers
 - Add two numbers
 - Find the product of two numbers

`pointer.c`

REVISION TIME!

STRINGS

- Strings are a collection of characters that are joined together
 - an array of characters!
- There is one very special thing about strings in C - it is an array of characters that finishes with a
 - This symbol is called a null terminating character
- It is always located at the end of an array, therefore an array has to always be able to accomodate this character
- It is not displayed as part of the string
- It is a placeholder to indicate that this array of characters is a string
- It is very useful to know when our string has come to an end, when we loop through the array of characters

HOW DO WE DECLARE A STRING?

WHAT DOES IT LOOK LIKE VISUALLY?

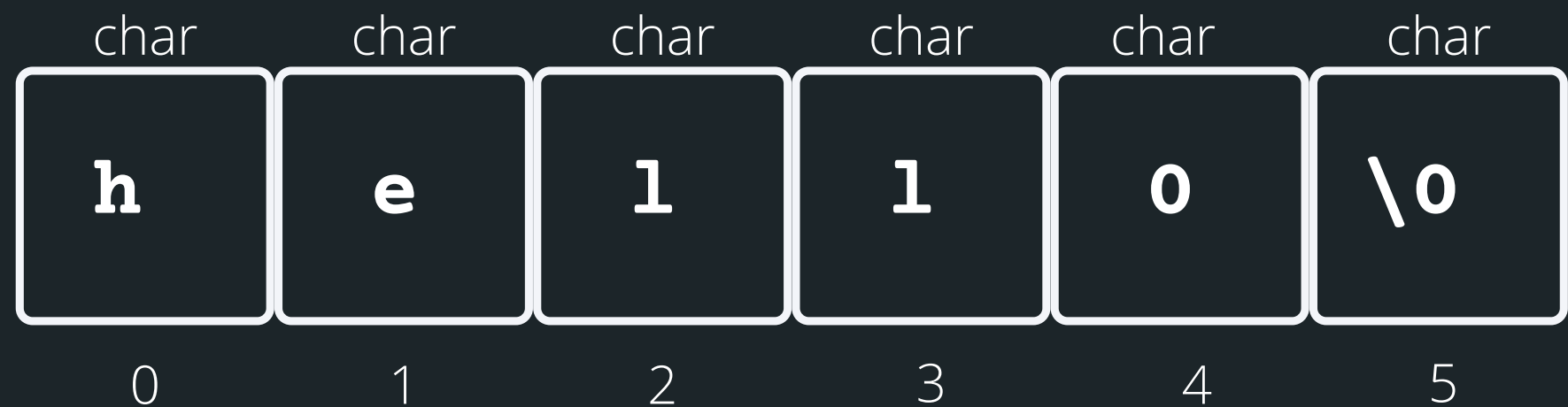
- Because strings are an array of characters, the array type is char.
- To declare and initialise a string, you can use two methods:

//the more convenient way

```
char word[] = "hello";
```

//this is the same as '\0':

```
char word[] = {'h', 'e', 'l', 'l', 'o', '\0'};
```



HELPFUL LIBRARY FUNCTIONS FOR STRINGS

FGETS()

There is a useful function for reading strings:

```
fgets(array[], length, stream)
```

The function needs three inputs:

- `array[]` - the array that the string will be stored into
- `length` - the number of characters that will be read in
- `stream` - this is where this string is coming from - you don't have to worry about this one, in your case, it will always be `stdin` (the input will always be from terminal)

```
// Declare an array where you will place the  
string that you read from somewhere
```

```
char array[MAX_LENGTH];
```

```
// Read in the string into array of length  
MAX_LENGTH from terminal input
```


```
fgets(array, MAX_LENGTH, stdin)
```

HOW DO I KEEP READING STUFF IN OVER AND OVER AGAIN?

Using the **NULL** keyword, you can continuously get string input from terminal until Ctrl+D is pressed

- `fgets()` stops reading when either length-1 characters are read, newline character is read or an end of file is reached, whichever comes first

```
1 #include <stdio.h>
2
3 #define MAX_LENGTH 15
4
5 int main(void) {
6     // Declare an array where you will place the string
7     char array[MAX_LENGTH];
8
9     printf("Type in a string to echo: ");
10    // Read in the string into the array until Ctrl+D is
11    // pressed, which is indicated by the NULL keyword
12    while (fgets(array, MAX_LENGTH, stdin) != NULL) {
13        printf("The string is: \n");
14        printf("%s", array);
15        printf("Type in a string to echo: ");
16    }
17    return 0;
18 }
```



LET'S PLAY!

Write a program that will read in a string from standard input and then count the frequency of each character that is in that string....

```
avas605@vx06:~$ ./string
Enter a string: this is the most awesome course
These are the frequencies of characters in the word this is the most awesome course

a occurs 1 times
c occurs 1 times
e occurs 4 times
h occurs 2 times
i occurs 2 times
m occurs 2 times
o occurs 3 times
r occurs 1 times
s occurs 5 times
t occurs 3 times
u occurs 1 times
w occurs 1 times
avas605@vx06:~$ ./string
Enter a string: ice cream
These are the frequencies of characters in the word ice cream

a occurs 1 times
c occurs 2 times
e occurs 2 times
i occurs 1 times
m occurs 1 times
r occurs 1 times
```

YOUR TURN TO PLAY :)

Write a program to take in a string from user and remove the first occurrence of a given character from that string.

```
avas605@vx07:~$ gcc string2.c -o string2
avas605@vx07:~$ ./string2
Enter string to scan in: I love COMP1511
Enter character to remove: C
After removing character, the string is: I love OMP1511
```

SOME OTHER INTERESTING STRING FUNCTIONS

<STRING.H>
STANDARD LIBRARY

CHECK OUT THE REST OF THE FUNCTIONS:
[HTTPS://WWW.TUTORIALSPOINT.COM/
C_STANDARD_LIBRARY/STRING_H.HTM](https://www.tutorialspoint.com/c_standard_library/string_h.htm)



Some other useful functions for strings:

- **strlen()** gives us the length of the string (excluding the '\0')
- **strcpy()** copy the contents of one string to another
- **strcat()** attach one string to the end of another (concatenate)
- **strcmp()** compare two strings
- **strchr()** find the first or last occurrence of a character

THE EXAM

EXAMPLE QUESTION 2



Perform some computation on a linked list

Given a linked list, print the largest value in that list

Edit the function

```
int largest (struct node *head)
```

WHAT DID WE LEARN TODAY?

REVISION

Strings
string.c

REVISION

Pointers
pointer.c

REVISION

Array of structs
direction.c

REACH OUT



CONTENT RELATED QUESTIONS

Check out the forum



ADMIN QUESTIONS

cs1511@unsw.edu.au