COMP1511 PROGRAMMING FUNDAMENTALS

LECTURE 3

Getting harder...

More complex IF statements,
A closer look at scanf(),
Breaking things, and
Learning about STRUCTS

LAST LECTURE...

LAST WEEK, WE TALKED:

- Welcome and Introductions
- Started looking at C
- Our first Hello! program
- Compiling and running your code
- **`printf()`** and **`scanf()`**
- Variables (.**`int`**, **`double`**, **`char`**)
- Maths :)
- Basic IF statements

IN THIS LECTURE...

TODAY...

- More complex IF statements
- Logical Operators
- Chaining **`if`** and **`else`**
- Breaking things
- Testing things

WHERE IS THE CODE?

Live lecture code can be found here:

HTTPS://CGI.CSE.UNSW.EDU.AU/~CS1511/22T1/LIVE/WEEK02/

HOW DO WE ASK GOOD QUESTIONS?

RELATIONAL OPERATORS

NOTICE: IN C, WE HAVE == AND =

THESE ARE NOT THE SAME AND DO NOT MEAN WHAT YOU ARE USED TO IN MATHS!

USING = WHEN YOU ASSIGN VALUES
USING == WHEN YOU ARE CHECKING FOR EQUIVALENCE

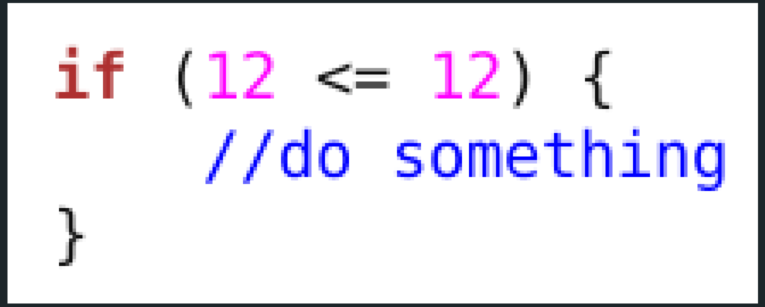
- Relational Operators work with pairs of numbers:
  - < less than
  - > greater than
  - <= less than or equal to
  - >= greater than or equal to
  - == equals
  - != not equal to

- All of these will result in 0 if false and a 1 if true

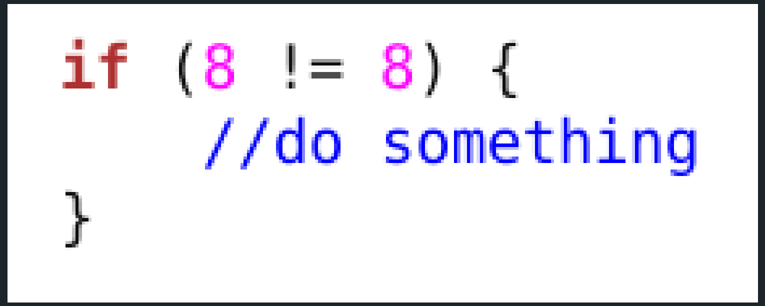
SOME EXAMPLES

LET'S TRY THIS OUT...

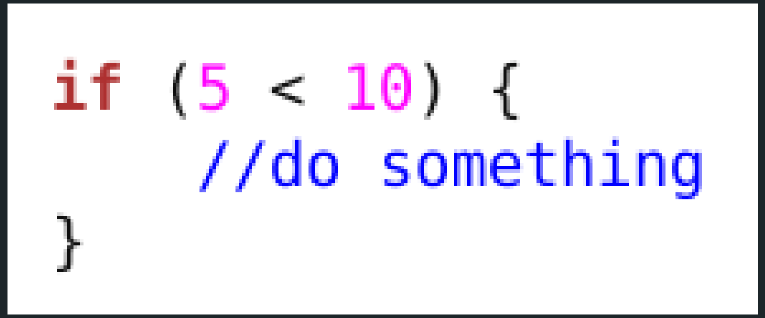
- True (1) or False (0)?



```
if (12 <= 12) {
    //do something
}
```



```
if (8 != 8) {
    //do something
}
```



```
if (5 < 10) {
    //do something
}
```

I LIKE QUESTIONS, HOW DO I ASK TWO QUESTIONS AT THE SAME TIME?

LOGICAL OPERATORS

The first two are used between two questions (expressions):

- AND: if both expressions are true then the condition is TRUE (equates to 1 if both sides equate to 1)
- OR: if any of the two expressions are true then the condition is TRUE (is 1 if either side is 1)

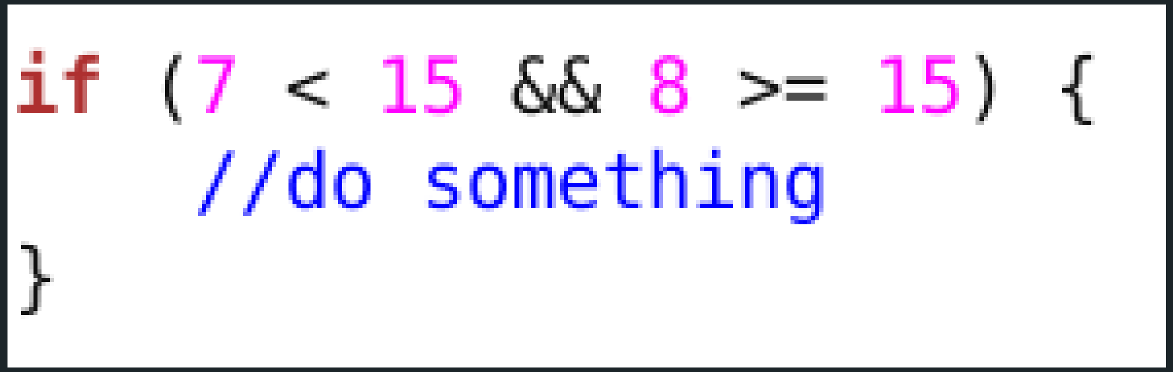
This is used in front of an expression:

- NOT: reverse the expression (is the opposite of whatever the expression was)

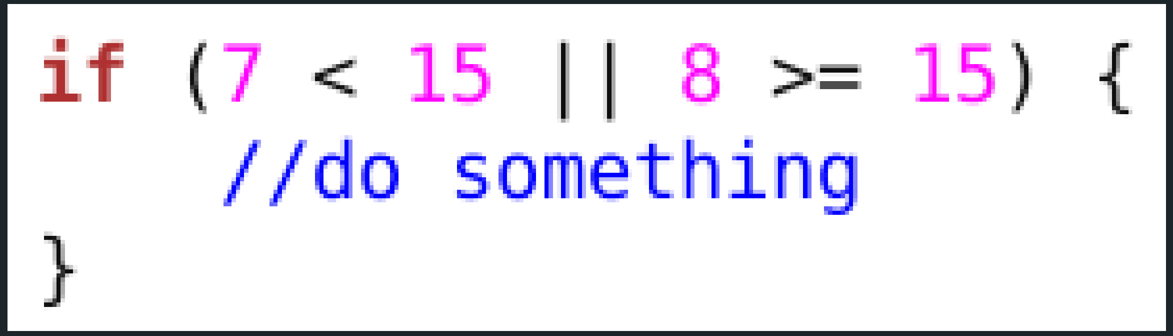
SOME EXAMPLES

LET'S TRY THIS OUT...

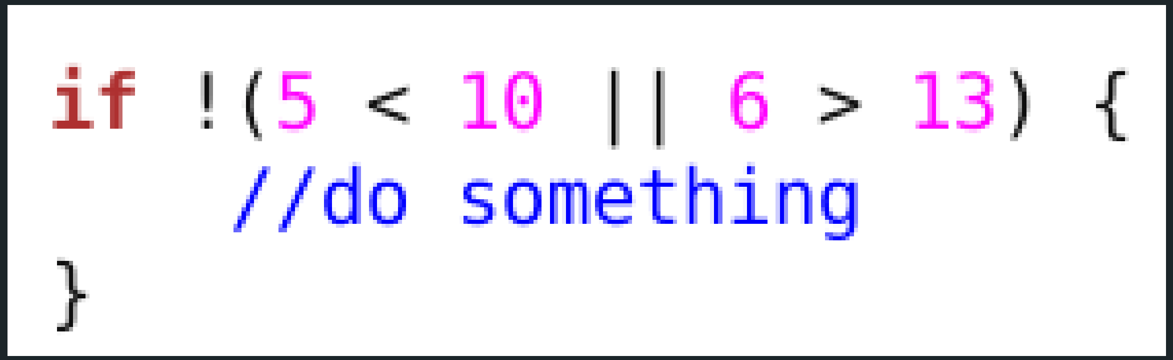
- True (1) or False (0)?



```
if (7 < 15 && 8 >= 15) {
    //do something
}
```



```
if (7 < 15 || 8 >= 15) {
    //do something
}
```



```
if !(5 < 10 || 6 > 13) {
    //do something
}
```

LET'S PUT OUR SKILLS TO THE TEST

LET'S CODE! (SOLVE THE PROBLEM FIRST)

A user rolls two dice and tell us the number on each of the rolled die. Our program will add the die numbers together and check them against a target number that only the program knows. It will then report back whether the total of the dice was higher, equal or lower than the secret number.

BREAKING DOWN THE PROBLEM INTO A SUM OF SIMPLE PARTS

A user rolls two dice and tell us the number on each of the rolled die. Our program will add the die numbers together and check them against a target number that only the program knows. It will then report back whether the total of the dice was higher, equal or lower than the secret number.

1. A user will roll two dice - done outside of our program
2. Take in the result of each die - how do we read input?
3. Add the die numbers together
4. Check them against a target number - based on steps 4 and 5, it looks like we need to make a decision - therefore IF statement
5. Output if total of the dice was higher, equal or lower than the target number - output based on the decision that we made

BREAKING DOWN THE PROBLEM INTO A SUM OF SIMPLE PARTS

A user rolls two dice and tell us the number on each of the rolled die. Our program will add the die numbers together and check them against a target number that only the program knows. It will then report back whether the total of the dice was higher, equal or lower than the secret number.

1. Take in the result of each die - how do we read input?
    a. Read input of die 1
    b. Read input of die 2
2. Add the die numbers together
    - sum = die1+die2
3. Check them against a target number - based on steps 3 and 4, it looks like we need to make a decision - therefore IF statement
    - Define the target number
4. Output if total of the dice was higher, equal or lower than the target number. - output based on the decision that we made
    - Is sum greater than target number?
    - Is sum less than target number?
    - Is sum equal to the target number?

NOW LET'S CODE!

1. Switch over toVLab
2. Open Terminal
3. Open a new file:

```
gedit dice_checker.c &
```

Feel free to follow along with lecture coding, or you can also find the code here:

IF / ELSE IF / ELSE

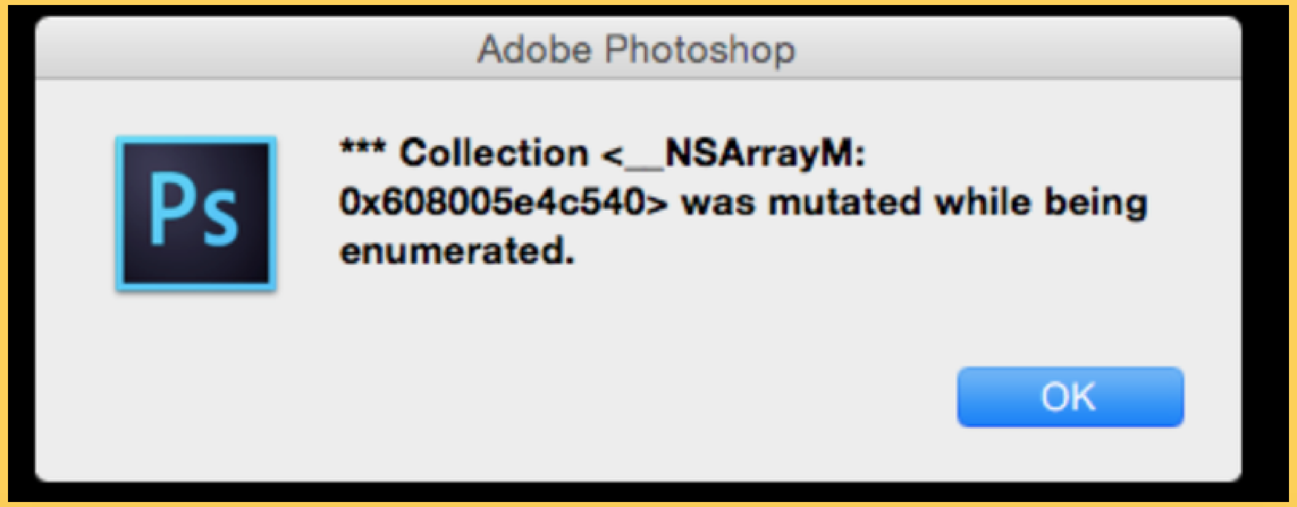
LET'S LOOK AT SOME CODE AND A DEMO

- IF statements with logical operators: **`if_logic.c`**
- IF statements with char:

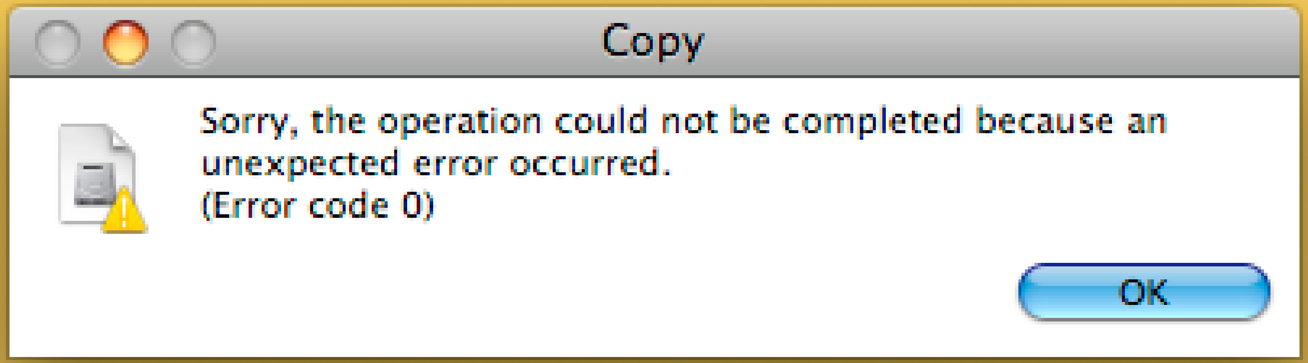
  `lower.c`
- Harder IF logic and chaining if and else together:

  `dice_checker.c`

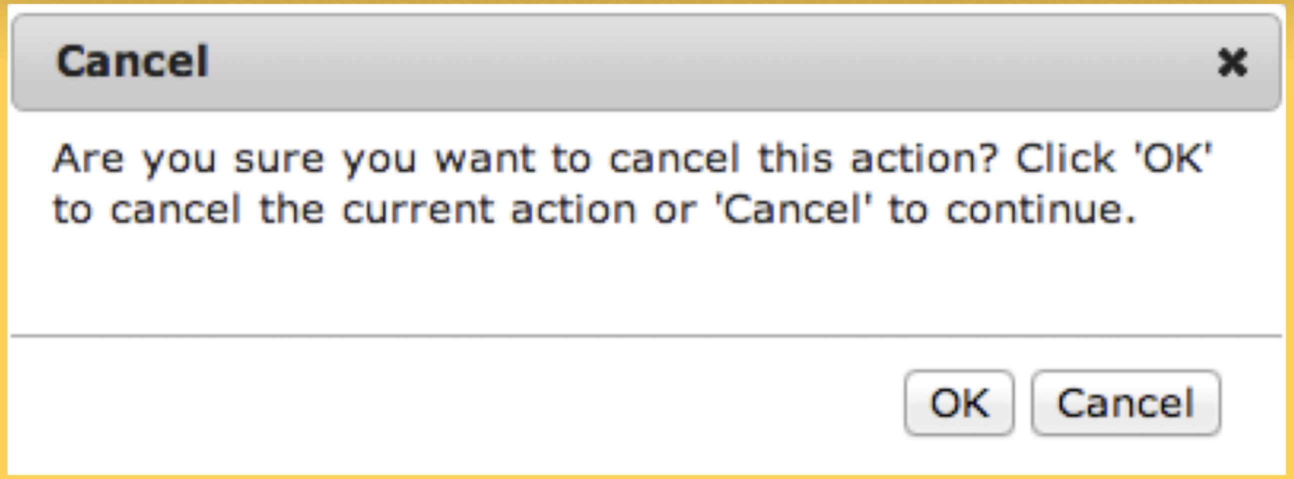
BREAKING THINGS



*** Collection <__NSArrayM: 0x608005e4c540> was mutated while being enumerated.



Sorry, the operation could not be completed because an unexpected error occurred.
(Error code 0)

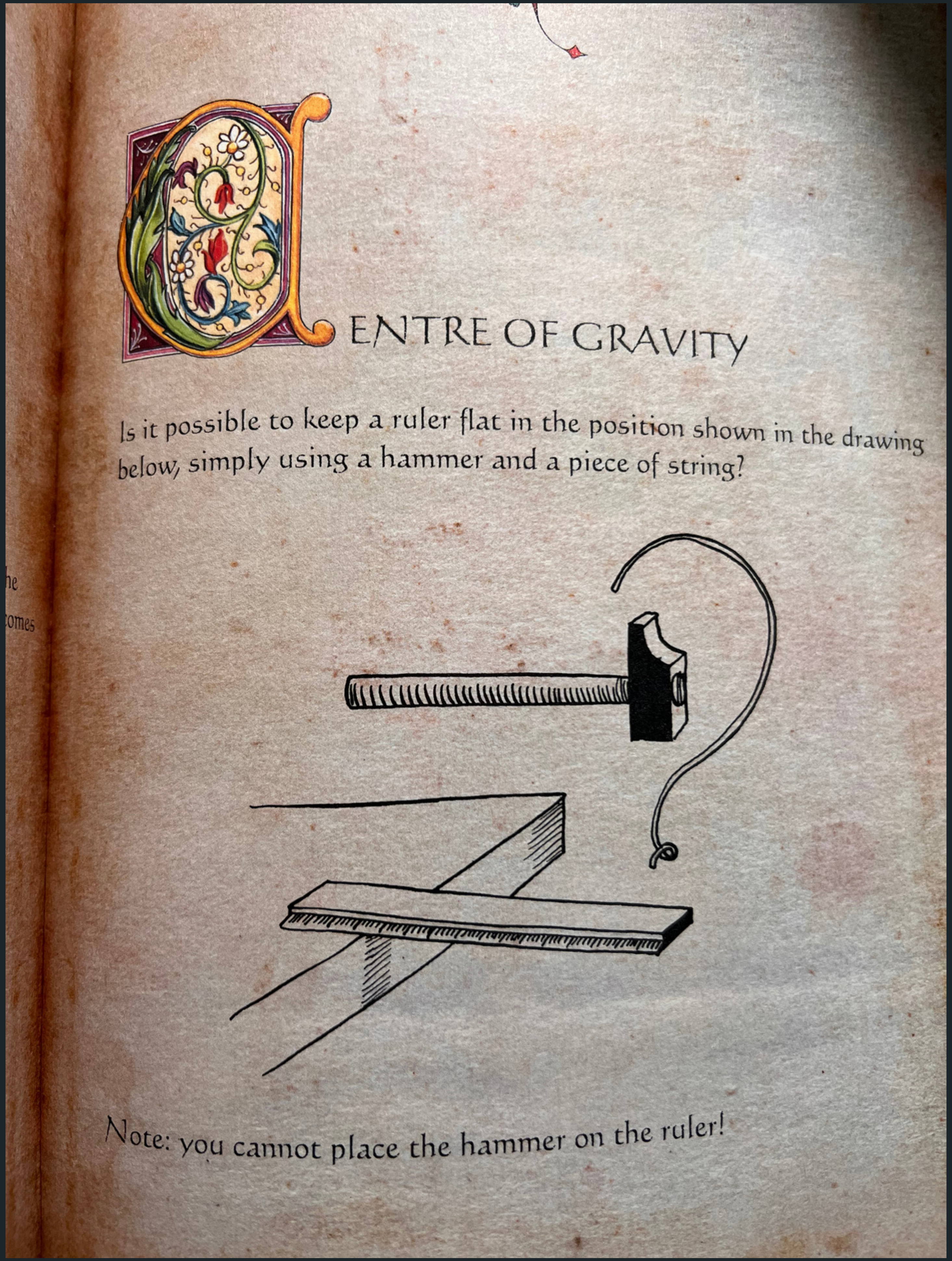


Are you sure you want to cancel this action? Click 'OK' to cancel the current action or 'Cancel' to continue.

It is really good practice to think about how it is possible to break your code? What can go wrong?

- Try and counter for these breaks!
- Important to have good error messages:
  - Tells the user exactly what has gone wrong
  - How can they fix it?
  - What is happening!?

BREAK TIME



CENTRE OF GRAVITY

Is it possible to keep a ruler flat in the position shown in the drawing below, simply using a hammer and a piece of string?

Note: you cannot place the hammer on the ruler!

HOW DOES SCANF() REALLY WORK?

A MAGICAL POWER...

- Gives us the ability to scan stuff in from the terminal (standard input)
- We have to tell the computer what we expect to scanf() - is it an **`int`**, **`double`**, or **`char`** ?
- But since scanf() is a function does it return something?
    - Yes, scanf() returns the number of input values that are scanned
    - If there is some input failure or error then it returns EOF (end-of-file) - we will look at this more tomorrow!
    - This is useful to check for any errors

DID YOU NOTICE HOW A NEW LINE IS READ BY SCANF()?

BECAUSE /N IS A CHARACTER ON THE ASCII TABLE: 10 LF (LINE FEED)

- You may have noticed that scanf("%d", &number) is able to ignore anything other than a number when it scans in - this is because whitespace is not a number and the function looks for a number
- But did you notice that this is not the case for

```
scan("%c", &character);
```

- This is because a new line (/n) is a character on the ASCII table, which means it is still a valid character to scan in (It is number 10 LF if you are interested!)
- To fix this, we can tell scanf() to ignore all preceeding whitespace by using a special magic trick:

```
scan(" %c", &character);
```

ORGANISING DIFFERENT TYPES INTO ONE RELATED WHOLE

USER DEFINED DATA TYPE `struct`

- Structures…. Or **`struct`** (as they are known in C!)
- Structs (short for structures) are a way to create custom variables
- Structs are variables that are made up of other variables

STRUCTURES

WHAT? WHY? EXAMPLES?

- What happens if you wanted to group some variables together to make a single structure?
- Why do we need structures?
    - Helps us to organise related but different components into one structure
    - Useful in defining real life problems
- What are some examples in real life where some things go together to make a single component?

HOW DO WE CREATE A STRUCT?

To create a struct, there are three steps:

1. Define the struct (outside the main)
2. Declare the struct (inside your main)
3. Initialise the struct (inside your main)

1. DEFINING A STRUCT

WHAT AM I GROUPING TOGETHER INTO ONE WHOLE? LET'S USE AN EXAMPLE OF A COORDINATE POINT

Because structures are a variable that we have created, made up of components that we decided belong together, we need to define what the struct (or structure is). To define a struct, we define it before our main function and use some special syntax.

```
struct struct_name {
      data_type variable_name_member;
      data_type variable_name_member;
      ...
   };
```

For example, using the coordinate point example, to declare a variable, cood_point, of type struct coordinate

1. DEFINING A STRUCT

WHAT AM I GROUPING TOGETHER INTO ONE WHOLE? LET'S USE AN EXAMPLE OF A COORDINATE POINT

For example, using the coordinate point example, to make a structure called coordinate, that has two members - the x_coordinate and the y_coordinate:

```
struct coordinate {
        int x_coordinate;
        int y_coordinate;
   };
```

2. DECLARING A STRUCT

INSIDE YOUR MAIN

To declare a struct, inside the main function (or wherever you are using the structure - more on this later)...

```
struct struct_name variable_name;
```

For example, using the coordinate point example, to declare a variable, cood_point, of type struct coordinate

```
struct coordinate cood_point;
```

3.INITIALISE A STRUCT

INSIDE YOUR MAIN

We access a member by using the dot operator .

```
variable_name.variable_name_member;
```

For example, using the coordinate point example, with variable name: cood_point, trying to access the x coordinate:

```
cood_point.x_coordinate;
```

LET'S SEE IT ALL TOGETHER FOR A COORDINATE POINT

1. DEFINE
2. DECLARE
3. INITIALISE

1. DEFINE

Inside the main function

```
// Define a structure for a
coordinate point

struct coordinate {
     int x_coordinate;
     int y_coordinate;
 };
```

2. DECLARE

Inside the main function

```
// Declare structure with
variable name

struct coordinate cood_point;
```

3. INITIALISE

Inside the main function

```
// Access stuct member to
assign value

cood_point.x_coordinate = 3;
cood_point.y_coordinate = 5;
```

LET'S SEE STRUCTS IN ACTION

CODE DEMO

You can see structs in action (I feel like we are in some sort of epic film here):

`struct_intro.c`

Feedback Please

I value your feedback and use to pace the lectures and improve your overall learning experience. If you have any feedback from today's lecture, please follow the link below. Please remember to keep your feedback constructive, so I can action it and improve the learning experience .

https://www.menti.com/m7h52ab7av

WHAT DID WE LEARN TODAY?

LOGICAL OPERATORS AND IF WITH CHAR

upper.c

CHAINING IF/ELSE AND ERROR CHECKING

dice_checker

TESTING

what should I test my code with?

SAY HELLO TO STRUCTS

struct_intro.c

REACH OUT

CONTENT RELATED QUESTIONS

Check out the forum

ADMIN QUESTIONS

cs1511@cse.unsw.edu.au