Need to know

- **Type casting**
- **Selection- If statement, nested if statement, if else, nested if else, switch case (c + pseudo code)**

Advantages of C:

- It easy to learn basic
- Syntactically similar to many other languages
- Programs written very fast
- Relatively portable

Disadvantages of C:

- C imposes very few restrictions: doesn't stop you making your own mistakes
  - Since it has very few restrictions writing good and reliable programs requires discipline
- Isn't as portable as some languages eg: java
- To use well you need good understanding of technical aspects of computer operations

Incremental assignment: how many times should happened we need to initialize variable as 0. int students = 0; if sensor goes of we add 1 to students. Counter is 0.

- int count = 0;
- count = count + 1
- count++
- count = count -1
- count--

Other assignment opperators:

- num1 += num2 : num1 = num1 + num2
- num1 -= num2: num1 = num1 – num2
- num1 *= num2: num1 = num1 * num2
- num1 /= num2: num1 = num1/num2

Variable naming: should relate to description.

- Can't use reserve words
- Do not begin with number
- Don't contain punctuation characters except underscore

- **Floating point/Integer division:** If you are dividing two integers but want the result to be a float then you need to force one of the numbers to be a float. Division before typecast

Float = int/float or float/float. Floating point division is what our calculator uses. However:

Int = float/int. The float/int overrides the declared int variable

There are four data types that store variables:

- **Integer**: int store (whole) number types
- **Float**: float for floating-point numbers or fractional numbers
- **Char**: character data type for single alphanumerical numbers and symbolic characters
- **Const**: **(constants)** Capital letters. Must declare and initialise (give number). So assigns unchanging value to the label. EG: const float Pi = 3.14

- Char Data Type:
  - The char data stores characters but also alphabets, numbers, symbols to a memory location
  - Character storing is done via ASCII system, which is symbols are assigned a number and this number is stored by computer.
  - The capital letters and lower case letters have a different ascii number assigned. That's why capital letters/lower case letters matter
  - Computers can only store or process numbers

  The reason why we put "%*C" with scanf function is because there is an issue of some buffer. So at the end of format specifier ie- %d

  - Algorithm → Test data → Write code → Test data

  Assumption: Assume gives data of the correct type. No rounding of results required.

  - Declare constant before variables. Const float GST = 0.1

  Testing data method:

  - Identify a list of test data that covers the boundaries of the likely inputs that your program will have to deal with
  - Identify how you expect your program to respond to these to inputs
  - Work through your algorithm using a pen and paper putting this test data into algorithm and seeing how it behaves.
    - This is called deskchecking. If results from algorithm is correct and expected then algorithm correct
  - Apply the same test data to your program once you've written it: If you get the same results as your algorithm then program is correct

Test Table: is a common way of showing test output from program.

Test Description	Inputs	Expected Outputs	Algorithm Outputs	Program Success/Failure
…..				

Test description: what the test aims to show
Inputs: List of input data relevant to test
Expected outputs: What you believe the correct result for the test should be
Algorithms output: The results you get after putting inputs into algorithms (desk checking) not code
Program success/failure: does the output from program match the output from algorithm

When writing this assume no error checking when trying to predict output/algorithm. IE allow out of range

Commenting: Break down the sections of code so ie Addition of codes. What would you find confusing if you read it for the first time. Not too low-level Comment level above. Being with lower case and don't space variable into two words

Indentation: Ident after curly brackets

- Boolean operators:
  - **AND:** If both true then truth if both false fase- **&&**

a	***b***	***a AND b***
0	0	0
0	1	0
1	1	1
1	0	0

  - **OR**: True if either is true- **||**

a	***b***	***a OR b***
0	0	0
0	1	1

a	*b*	*a OR b*
1	1	1
1	0	1

- **XOR**: difference from or when both are true the answer: is false

a	*b*	*a XOR b*
0	0	0
0	1	1
1	1	0
1	0	1

- **NOT:** Complimentary operator inverses your operands: **!a**

x	*NOT x*
0	1
1	0

Data types: Integers (whole numbers) floats (fractional data) char (alphabet characters)

There are four data types that store variables:

- **Integer**: int store (whole) number types
- **Float**: float for floating-point numbers or fractional numbers
- **Char**: character data type for single alphabetical number and symbolic character
- **Const (constants)**: capital letters. Must declare and initialise. Assigns unchanging value to label eg: const float pi =3.14

BELOW ARE NOTES:

#include <stdio.h>: allows us to use the input/output routines that c provides.

Int main(): declaration of the main function (chunk of code that does something). First function.

{}: beginning symbol of a block of code.

/* */: this is a comment

printf(): Invokes the printf() function/calls to it.

Return 0: out of function you are in ie taking you outside main function which terminates the program.

Problem can be divided into three components.

- Input: list of source data provided by the problem (nouns/adjectives)
- Processing: a list of actions that need to be taken to produce an output (verbs). List of actions needed to be performed. Not how to peform.
- Output: a list of outputs required (nouns/adjective)

EG: that reads in three integer numbers from the user, computes the sum of the three numbers and reports the result

However, best find output if you can't find.

Pseduo code:

IF (Expression) THEN
    Value

Big note:

If there are no brackets in psuesdo code the first line gets done first

Type casting: converting a data type to another. The value only thats the important bit not the variable. Put the data type e in front of the variable. Eg: x = int (y). So taking the value of y and casting it to int eg: 2.6 → 2

To be specific the variable x is still the same. So if x = 3.2 type cast to float is 3 but if you call x it is 3