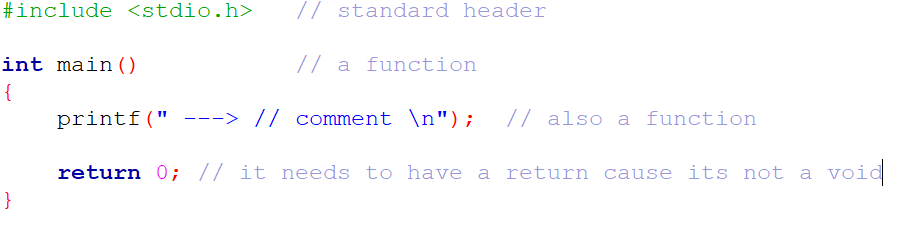
COMPUTER PROGAMMING

Note: You will learn C language until the finals no other sub topic other than this language, I will just put the simplified version of the learnings we learned from this subject.

Fundamentals in ©

This is the easiest to make but it has many things going on



Lets break it down:

**Standard header file**

* It is used for the code you are using to work

An example was if you use a code from another library the code will print an error in your code

**Other standard library**

* Stdio.h means standard input && output , contains most of the code you learn in this word doc
* Stdlib.h means standard library, contains the memory allocation (only in the course I learned this but it’s a waste of time I you want to speed tun this © language)
* String.h contains some helpful code like copy the length of the array + other helpful code
* Math.h contains math operation + calculation
* Ctype.h contains the code to make the inputs user upper or lower case and more
* Stdbool.h contains the 0 && 1 means true or not code

printf(“Lets skip the use of function for now cause you need to learn it later”);

printf – print the letter || (means or) number inside “ ”

// ( 2 backslash ) means comment

/\*\*/ an upgraded backslash means you can store more code inside the \*.

syntax of c is simple its just ;

(lets make it pink so that you remember it better)

To note

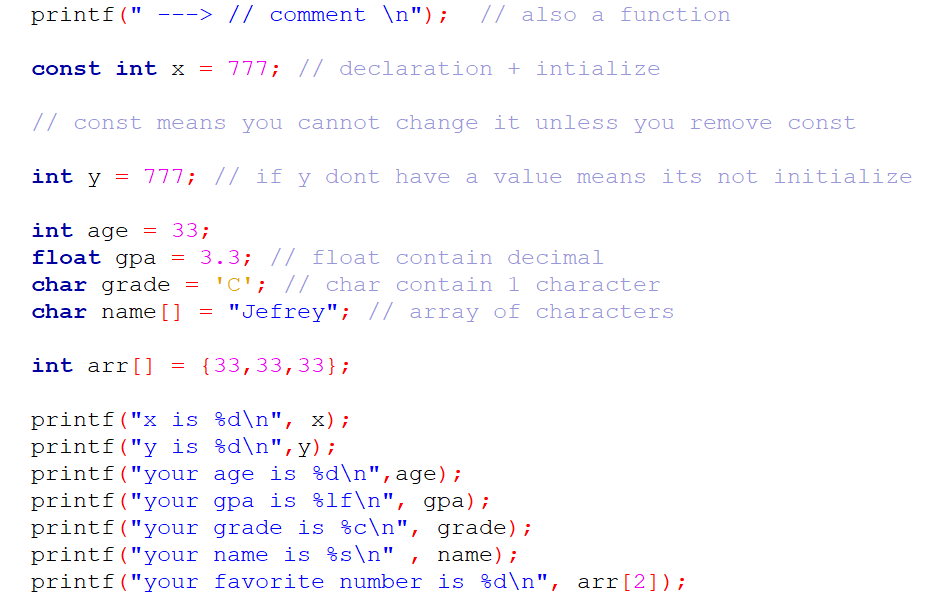
* © is a case sensitive language that means you should not wright ,this is a variable , (it will give it an error)

You should write variable declaration like this

thisIsNumberOne;

this\_is\_number\_one;

**Variable || Arrays || constants || data types**



The difficult part of learning variables is by naming it

const int x = 777;

int && float && double && char is from data types

* They contain the number of bytes from char being 1 and the high is the double.

const is just constant

x is the variable that is being paired with the data types

data type + name + [] is just the **array** of © then you can declare the number of bytes from the {} or “ ” .

To access the output of the data type you need a format specifier

- Int %d

- Char %c if it’s a word %s

- Float %f you can manipulate the decimal value by putting 2f to show the 2 decimal place only.

- double %lf better version of float

**char array[5] = {1,2,3}**

**the machine see it as 0,1,2 . why? It has a null character \0 at the end**

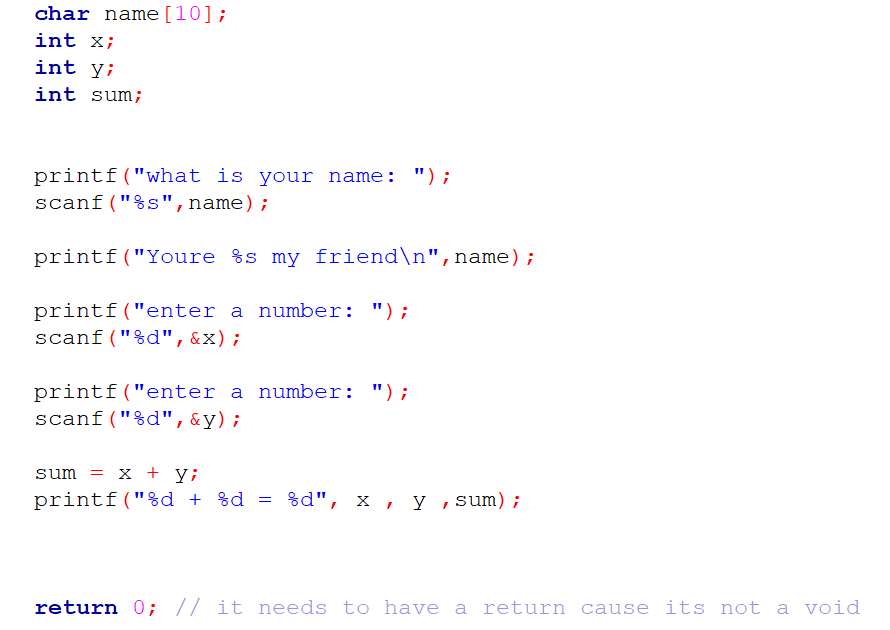
to print it from the printf function you need

printf("your age is %d\n",age);

place the variable that you name from the initialized data type and don’t forget the , comma.

\n means new line

**User input || arithmetic operators || logical operator**

****

**To get the users input you need scanf + the format specifier for the users input if it’s a character or a number and the ampersand (&) for the storage of this input then the name variable   
- scanf(“%d”, &x);**

**Arithmetic operators**

+ , - , \* , / and % means modulus it counts remainder

++ means increasing the number by 1

--(decrement) means decreasing the number by 1

**Logical operator is used in an if statement**

&& and : both must be true

|| or : either one is true

!= not: reverse if its true its false

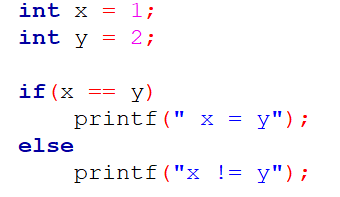
== equal

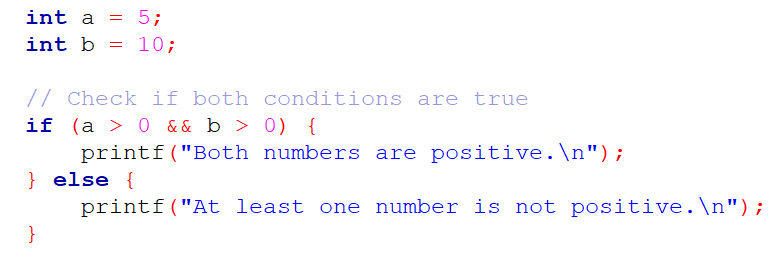
If statement has 2 parts the **else if** && **else**

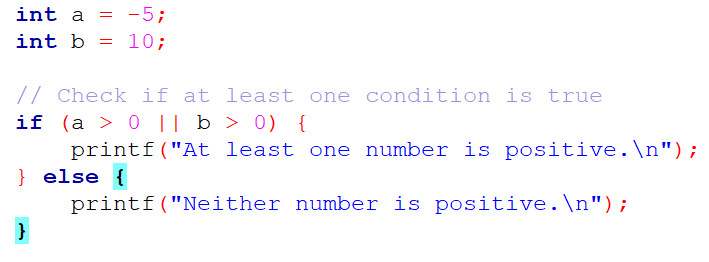
**If contains the condition in the (then you can put the logical operator inside here)**

**Else if executes if there is no true condition in the first statement**

**Else execute cause it’s the default option for the code or something else**

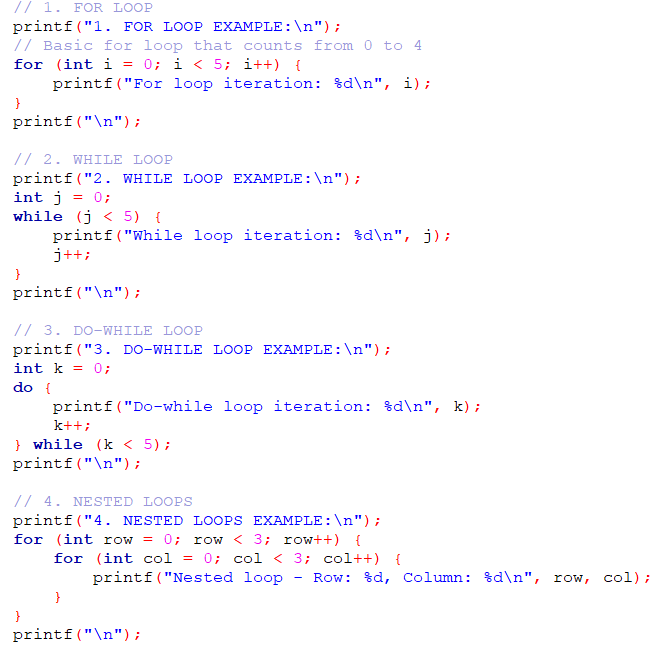
****

****

****

**Loops && switch**

**For Loops contain an index + a condition + increment how long do you want to keep this loop**

****

 **For Loop**: A traditional loop with initialization, condition, and increment.

* Syntax: for (initialization; condition; increment/decrement)
* Best used when you know the number of iterations in advance

 **While Loop**: Continues executing while a condition is true.

* Syntax: while (condition) { ... }
* Good when you don't know the exact number of iterations beforehand

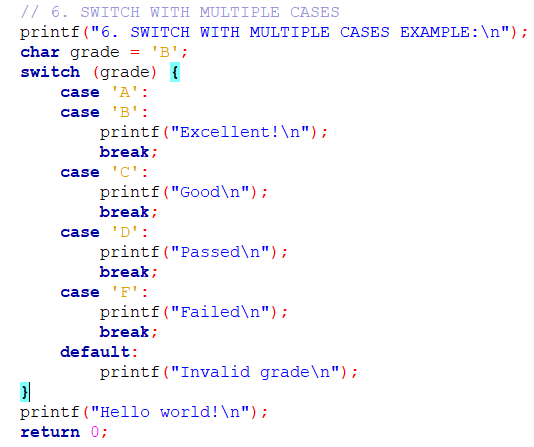
 **Do-While Loop**: Similar to while loop, but guarantees at least one execution.

* Syntax: do { ... } while (condition);
* Useful when you want the code block to run at least once

 **Nested Loops**: Loops inside other loops

* Demonstrates how to create multi-dimensional iterations





 **Switch Statement**: A multi-way decision structure

* Syntax: switch (expression) { case constant: ... break; }
* More readable alternative to multiple if-else statements
* Always include break to prevent fall-through
* default case handles any values not explicitly matched

**FINALS TOPIC**

**POINTERS || FUNCTIONS**

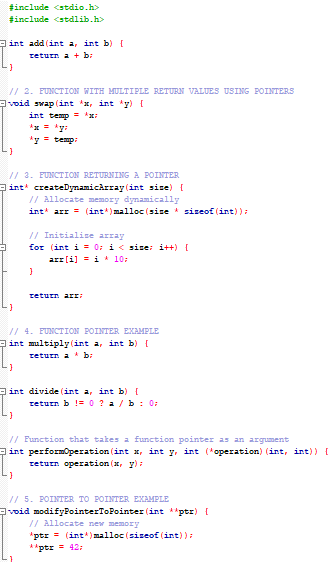
1. **Basic Functions**
   * Simple functions that take arguments and return values
   * Declared with return type, name, and parameter list
   * Use return to send back a value

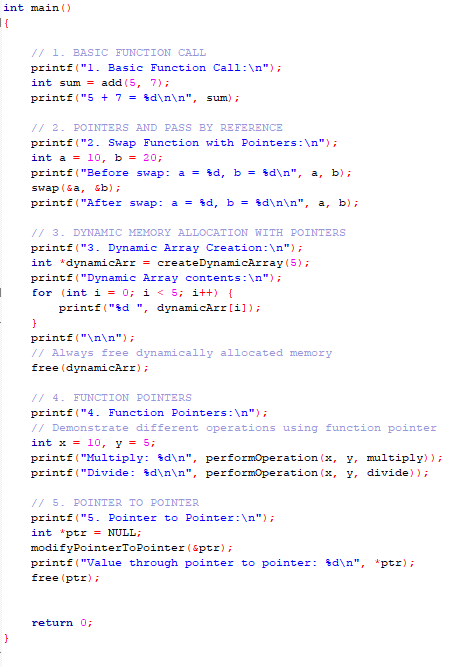
In other words its outside of the main function which makes the code

1. **Pointers and Pass by Reference**
   * Pointers allow you to modify original variables
   * swap() function demonstrates how to exchange values
   * Use & to get address, \* to dereference
   * Allows passing and modifying variables directly
2. **Dynamic Memory Allocation**
   * malloc() allocates memory dynamically
   * Always free() dynamically allocated memory to prevent memory leaks
   * Useful for creating arrays or structures with runtime-determined size
3. **Function Pointers**
   * Allows passing functions as arguments
   * Can store and call functions through pointers
   * Enables flexible, dynamic function calls
   * Syntax: return\_type (\*pointer\_name)(param\_types)
4. **Pointer to Pointer**
   * Additional level of indirection
   * Useful for more complex memory manipulations
   * Can modify pointer's target dynamically

Important Pointer Rules:

* Always initialize pointers
* Check for NULL before dereferencing
* Free dynamically allocated memory
* Be careful with memory management to avoid leaks





In short function is just outside of main and you use a function call inside the main to execute the function

You can use a function call from the if statement and inside the for loops and many more

if (function\_call)

for(int I = 0; I < 5 ; I ++ )  
{

function call here

}

For pointers you need an asterisk and p for the variable and never forget to specify the data types the int and char.

