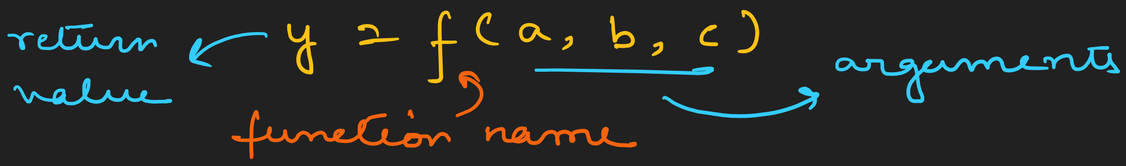
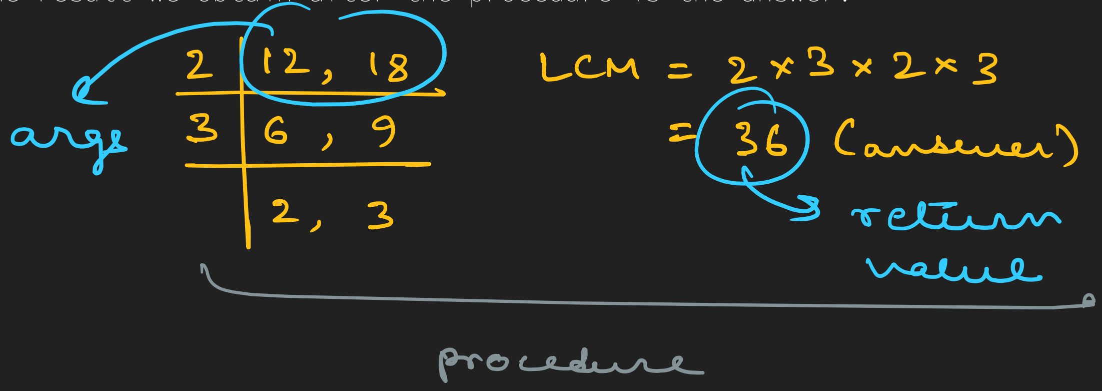
Functions in C.

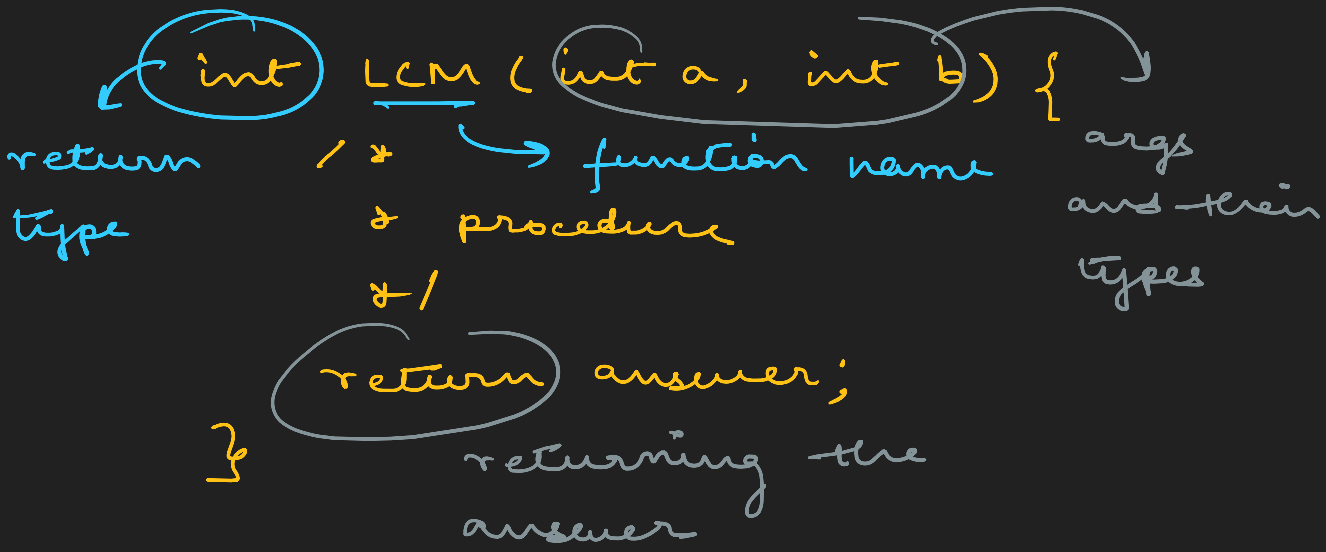
1. What are functions anyway? We have seen functions in Maths. A function takes some input (called arguments) and returns some output (called return value).



2. Take, for instance, the example of LCM. What will be LCM(12, 18)? LCM(12, 18) is read as LCM of 4 and 3. How do we solve it? We draw a table, write down the numbers, then find the common prime factors... What we follow is a procedure. And then the result we obtain after the procedure is the answer.



3. If we want to do the same thing in C. It will look like this:



```
int LCM ( int a, int b) {
    /*
     * procedure
     */
    return answer;
}
```

Types of functions.

1. Built-in functions - Functions that are available to us without including any library. There are no built-in functions in C, but in other languages like python, they provide a lot of functions which we can use without including any library, like print(), input(), round() and more.
2. Library functions - Other people have written functions to make our like easier, and we can just include them into our code and use them. Examples include printf() and scanf() from stdio.h and pow() and round() from math.h
3. User-defined functions - These are the functions that we create. The LCM() function we roughly outlined previously is one such example.

What type of function is the main() function?

1. int main() is a function very similar to our int LCM(int a, int b) function.
2. Both of their return type is int and we have to define the procedures ourselves in both of the functions, but unlike LCM(), main() is not a user-defined function.
3. main() is a special function that must be defined if we want to run our C program. void main() or int main() is the starting point of every C program.

What is the difference between int main() and void main()?

1. From the syntax itself, we can see that int main() returns an integer, while void main() returns nothing.
2. The integer that int main() returns is called the status code. When our program ends, the status code can be viewed by the user to check how the program ended.
3. We generally end our int main() function with a return 0. This means that no errors were encountered and everything went fine. But if we did face an issue, we could return 1 or any other non-zero integer to let the user know that something went wrong.
4. Status code in this case is similar to the status code we encounter while browsing the Internet. In the Internet’s case, a code of 200 means everything went fine, a code of 404 means no such page was found and a code greater than 500 means something went wrong in the server-side. Using a status code, we can convey different messages using just integers.
5. void main() does not explicitly return a status code. C returns a status code on its own depending on how our program ended.

Write a C function which returns the LCM of two integers without finding the GCD of the two numbers.

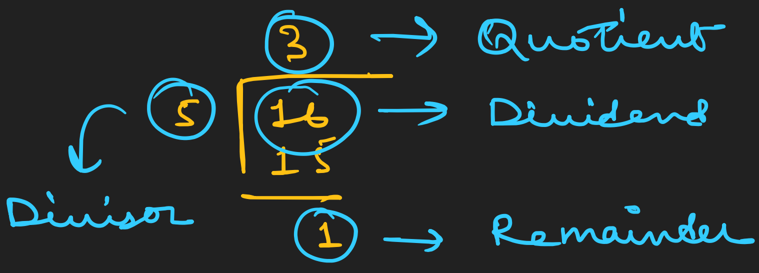
```
int LCM(int a, int b) {

      // What is LCM? The lowest multiple of A

      // which is also a multiple of B, i.e lowest

      // multiple of A which is divisible by B.

      // Note that A * B is guaranteed to be A

      // common multiple of A and B. So, we will

      // find our LCM between A * 1 and A * B.

      for (int i = 1; i < b; ++i) {

            if ((a * i) % b == 0)

                  return a * i;

      }

      // If A * i for i = 1 ... B-1 is not the LCM

      // then the LCM must be A * B.

      return a * b;

}
```

Analysing the above code.

1. Note that the for loop runs from 1 to B-1 in the worst case. So, it almost takes B steps to find the LCM. If B is a very big number, in the worst case, it will take a lot of steps.
2. We can optimize it further by choosing the smaller number as B, but both A and B may be very big numbers.
3. A better way of finding LCM would be by dividing the product of A and B by their GCD. But for that we again need to define a GCD function which should take less than B-1 steps.
4. Turns out that there is a very efficient and simple way to find GCD of two numbers. It is called the Euclidean method.

Euclidean method of finding GCD.

1. Before proceeding, we need to recall some terminologies related to division.



2. We divide the dividend by the divisor to obtain the quotient which leaves the remainder. That’s what’s going on when we are dividing.
3. Euclidean method is like division, except that, after dividing once, we replace the dividend with the divisor and the divisor with the remainder, and then divide again. This continues until the remainder is 0. The divisor for which remainder is 0 is the GCD.

Write a C function to find GCD of two numbers using Euclidean method.

```
int GCD(int a, int b) {

    // A is the dividend

    // B is the divisor

    // Note that A need not be greater than B.

    // If A is less than B, then in the next

    // iteration, A and B will be interchanged

    // anyway. I encourage you to do a dry run

    // and verify it.

    int r = a % b;

    while (r != 0) {

        a = b;

        b = r;

        // The order of assignment is very

        // important! If B = R were used before
        // A = B, then the new value of A would
        // be R and not B (which is not what we
        // want!)
        r = a % b;
    }
    return b;
}
```

Back to LCM.

1. Now that we know an efficient way to find GCD, we can use the new GCD() function to find GCD and then use it in another function LCM_better() to obtain LCM of two numbers. And that will exactly be the homework.
2. Another good assignment would be to reimplement LCM(), GCD() and LCM_better() but instead of returning the result, take an int * as argument and change the address itself to store the result.