**DATA STRUCTURES AND ALGORITHM**

**SELF-NOTES**

**Virendra Shinde (20BAI10132)**

Channel Name- Apna College

Link- https://www.youtube.com/watch?v=z9bZufPHFLU&list=PLfqMhTWNBTe0b2nM6JHCnAkhQRGiZMSJ

Data structures- arrangement of data so that they can be used efficiently in memory. Ex- arrays, linked lists, stack, queue.

Algorithms- sequence of steps on data using efficient data structures to solve a given problem.

There are three types of data types-

1. Primitive-

Integer, float, character and Boolean.

1. Derived-

Function, array, pointer and reference.

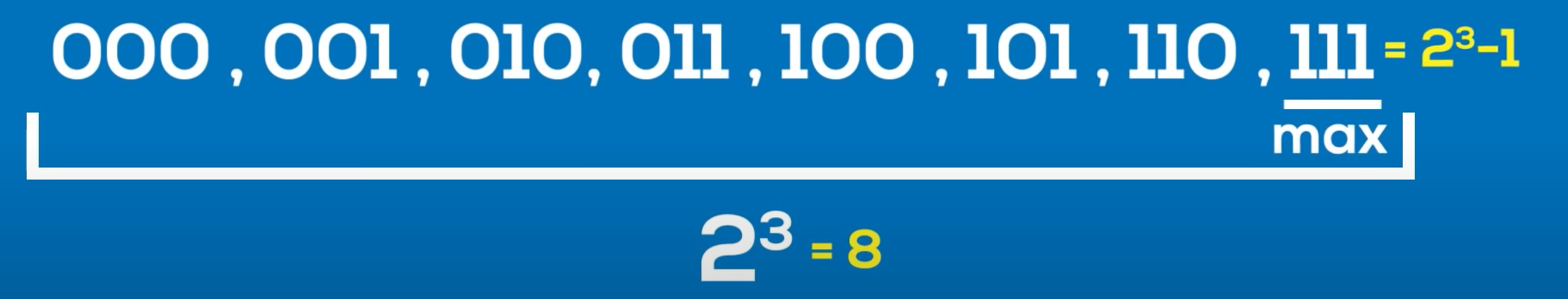
1. User-defined-

Class, structure, union and enum.

Any integer is stored in 4 bytes.

Whereas 1 byte is equal to 8 bits.

That means a number stored in a memory is consuming a space of 32 bits.

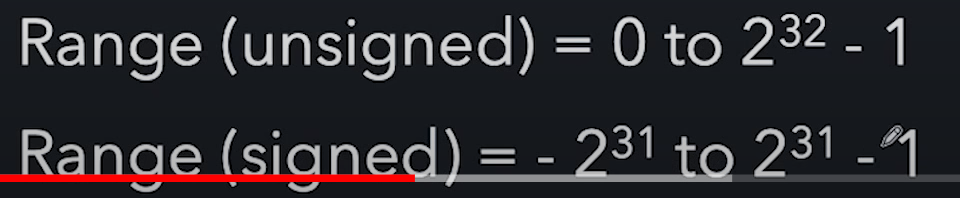


3 bits example.

The maximum number I can store here will be 2^32 – 1.

From all these 32 bits, the first bit is known as the most significant bit (MSB), because it tells whether a number is negative or positive.

If the first bit stores the value 1 then the number is negative and if the bit is stored with a 0 then we can tell that the number is positive.



**Float-**

Size= 4 byte

(Up to 7 decimal digits)

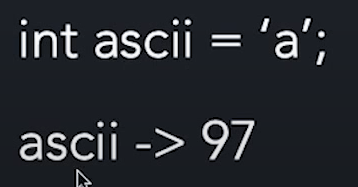
Even if we want to store a number with more than 7 decimal digits we can do that with the help of another storage data type that is DOUBLE.

Double has a storage/ size of 8 bytes and can store up to 15 decimal digits.

**CHAR**

Every character and a symbol are assigned with a dec, hex and oct number.

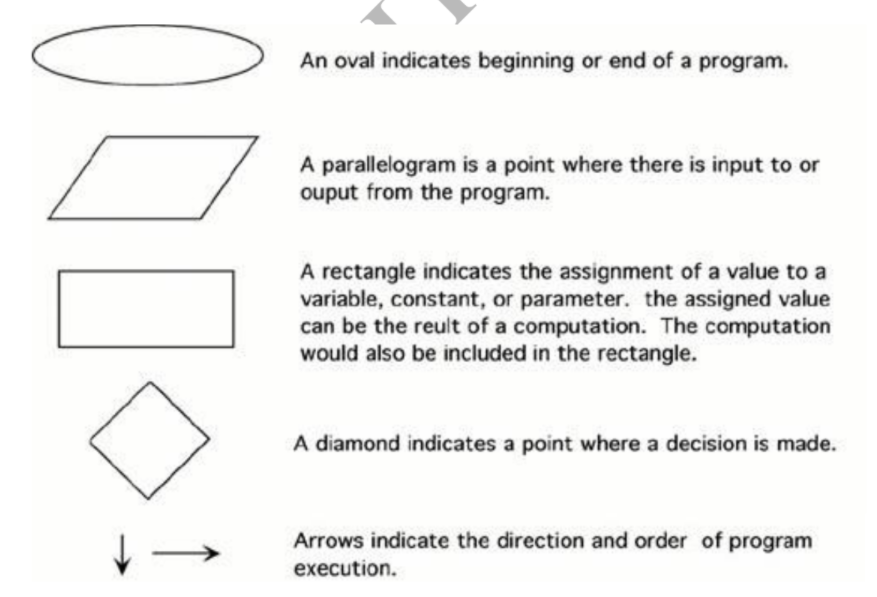
If we try to run-

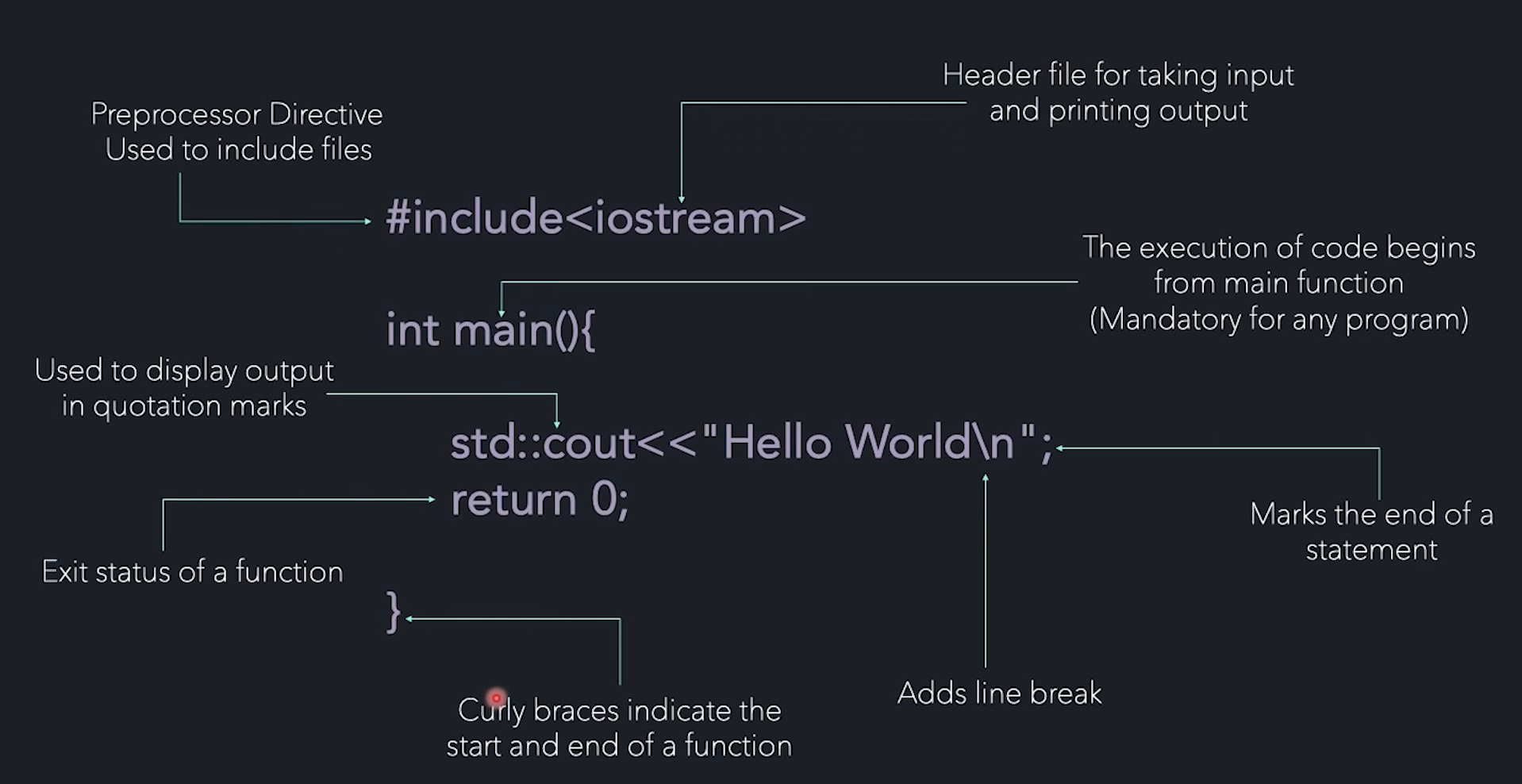


**BOOLEAN**

Size- 1 byte

It stores only True and False.





<< -insertion operator

>> -extraction operator

**If Else statement in C++**

Declaration

Initialization

Calling

**Continue-** Skips to the next iteration of the loop

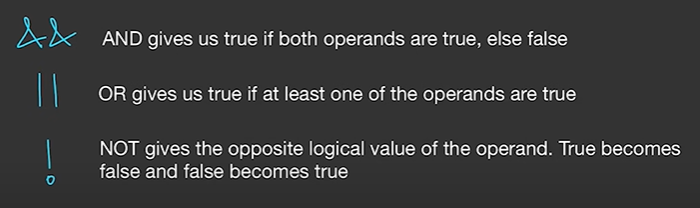
**Break-** Terminate the loop.

**Switch**

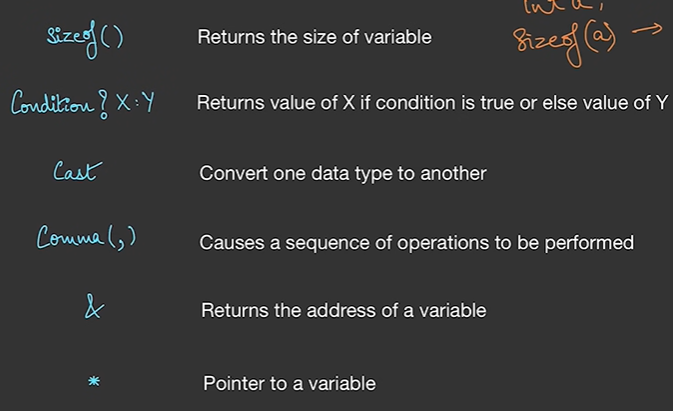
Switch can help us to reduce the codes for multiple if else if and else statements and make the code perform the same thing using switch case.

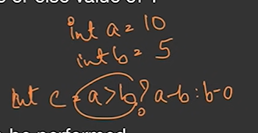
(Considered to be more efficient too!)

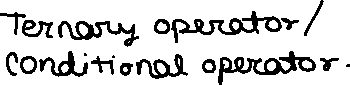
Operators are symbols that tell the compiler to perform some specific operations.



Xor- same values gives 0, and different values gives 1.







Binary system-

Functions are a specific code that performs a task.

Stack is a storage that stores the information regarding all the function may it be user defined or pre-defined.

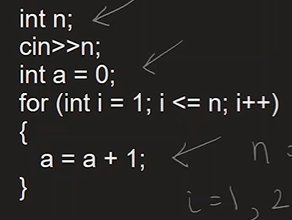
Single function information stored in a well mannered is called as a stack frame.

Multiple stack frames create a stack.

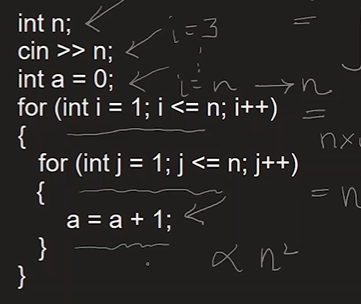
When we add any function or initiate a function it is called to be pushed in to the stack and when the work is done of the function and the function has successfully returned the desired output the function is then removed from the stack and called to be popped from the stack.

**Time Complexity-**

Time complexity of an algorithm quantifies the amount of time taken by a program to run as a function of length of the input.



For the very basic code as given above the amount of time taken by the algorithm to run is directly proportional to the amount that we have entered in the code as an input for the variable. {linearly proportional}

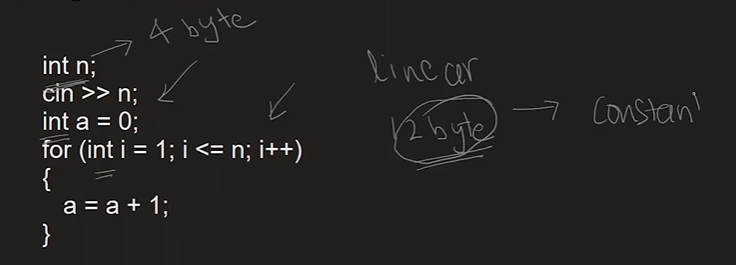


And for a bit complex program like this, the time is directly proportional to the square of the input entered in to the variable.

**Space Complexity**

A program that takes less time and moreover consumes less space/ memory to run and perform the given task, then the program is said to be efficient.

Space complexity of an algorithm quantifies the amount of time taken by a program to run as a function of length of the input. It is directly proportional to the largest memory your program acquires at any instance during run time.



The space is constant regardless of the input that is given to the program.

**Worst Case- [O big o notation]**

in this case, the variable is found at the last or some times the variable is not found in the whole list even if the list is over.

Most time taken to find the desired variable.

Time called as worst time complexity.

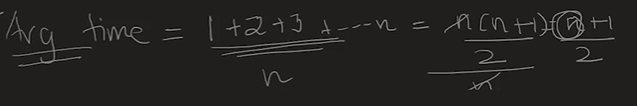
Time complexity is directly proportional to the size of the list.

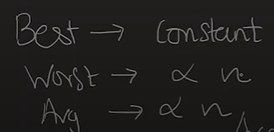
**Best Case- [big omega notation]**

in this case the variable/ value is found at the very first position.

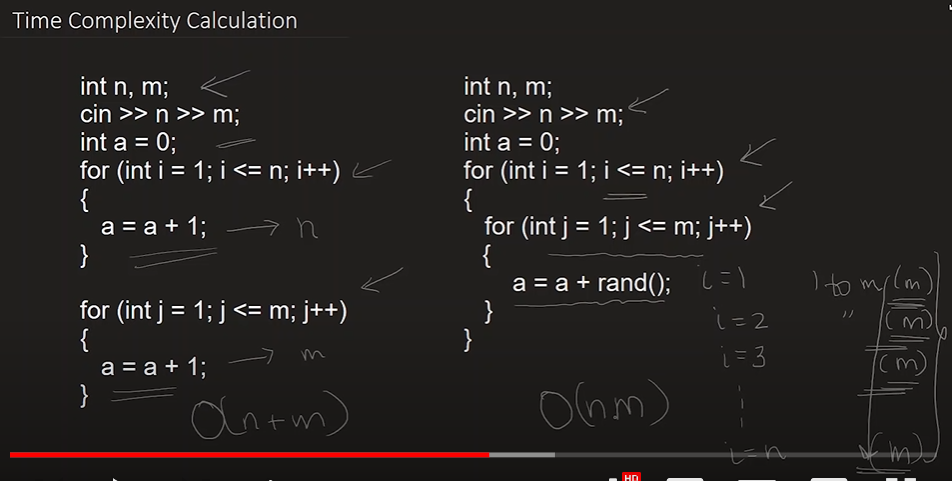
Time complexity is constant.

**Average Case- [big theta notation]**





Time complexity calculation-

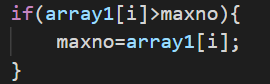


**Introduction to Arrays-**

Array is a list of variables of similar data types.

One value will consume 4 bytes so the total space/ memory consumed by the array will be 4\*(size of the array).

For including functions of c++ like INT\_MIN and INT\_MAX we need to add/ include header file called climits.



Instead of writing these 3 lines of code to find max number we can alter this with-



This pre-defined function in c++ compares two values given in ()

And stores the max value into the variable maxno.

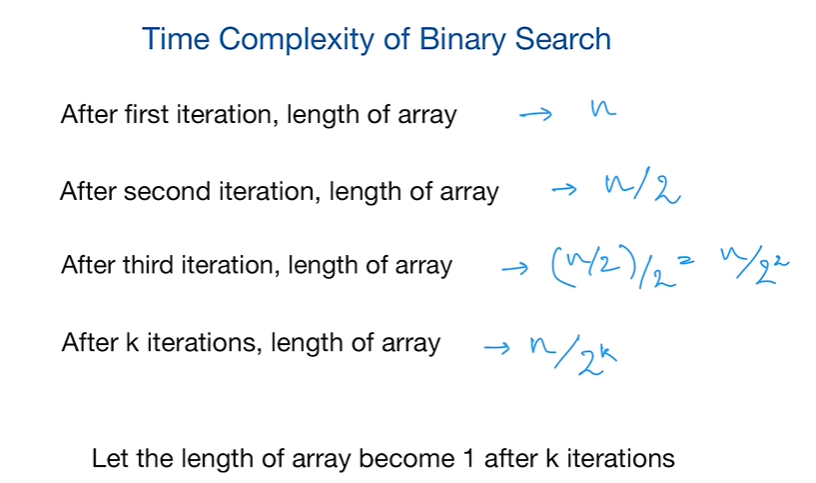
**Searching in Arrays**

1. Linear search

Time complexity – O(n)

1. Binary search

Mid point -> checking whether less or greater -> changing the start and end point.

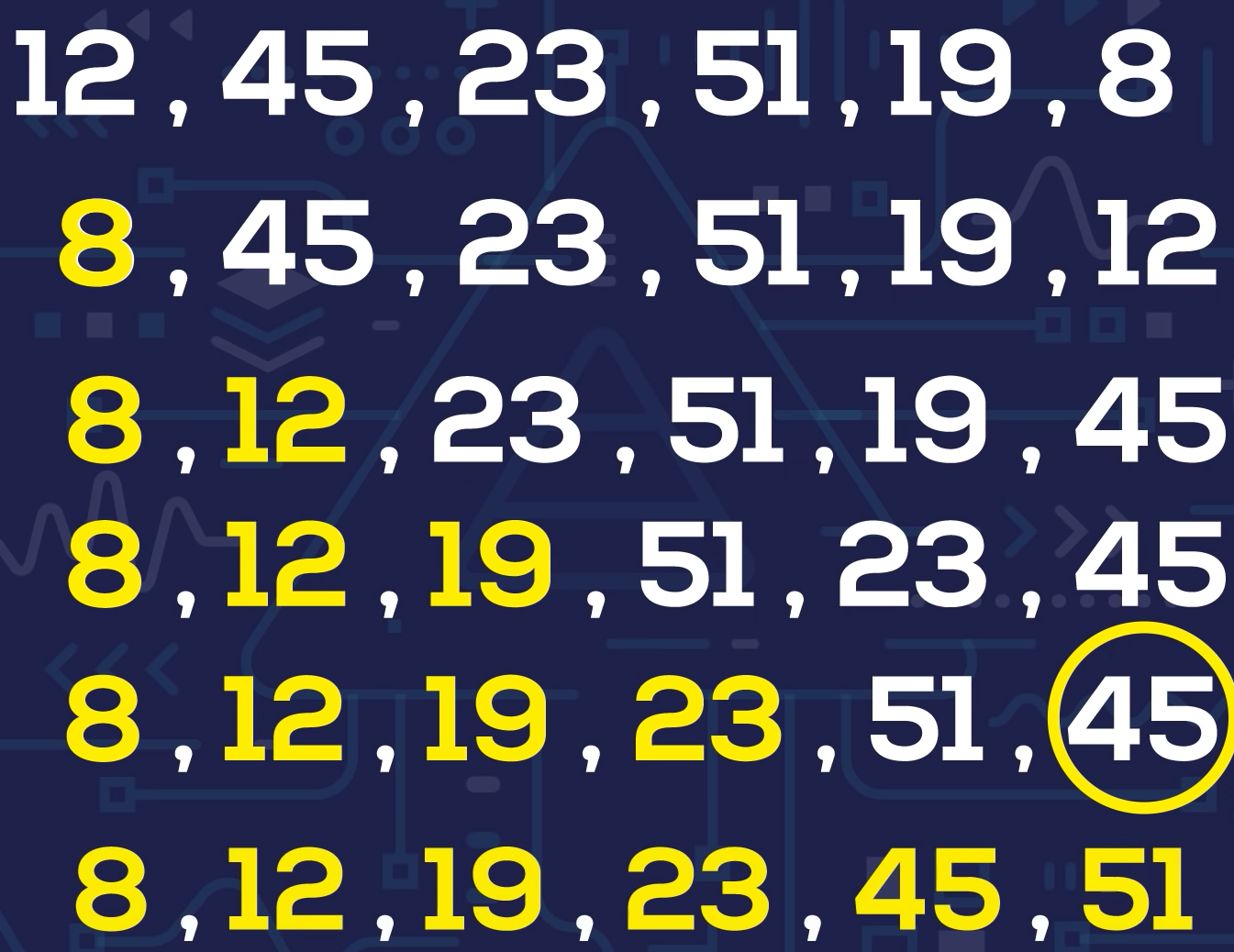


**SORTING IN ARRAYS**

Sorting is arranging the elements in an array in ascending or descending order.

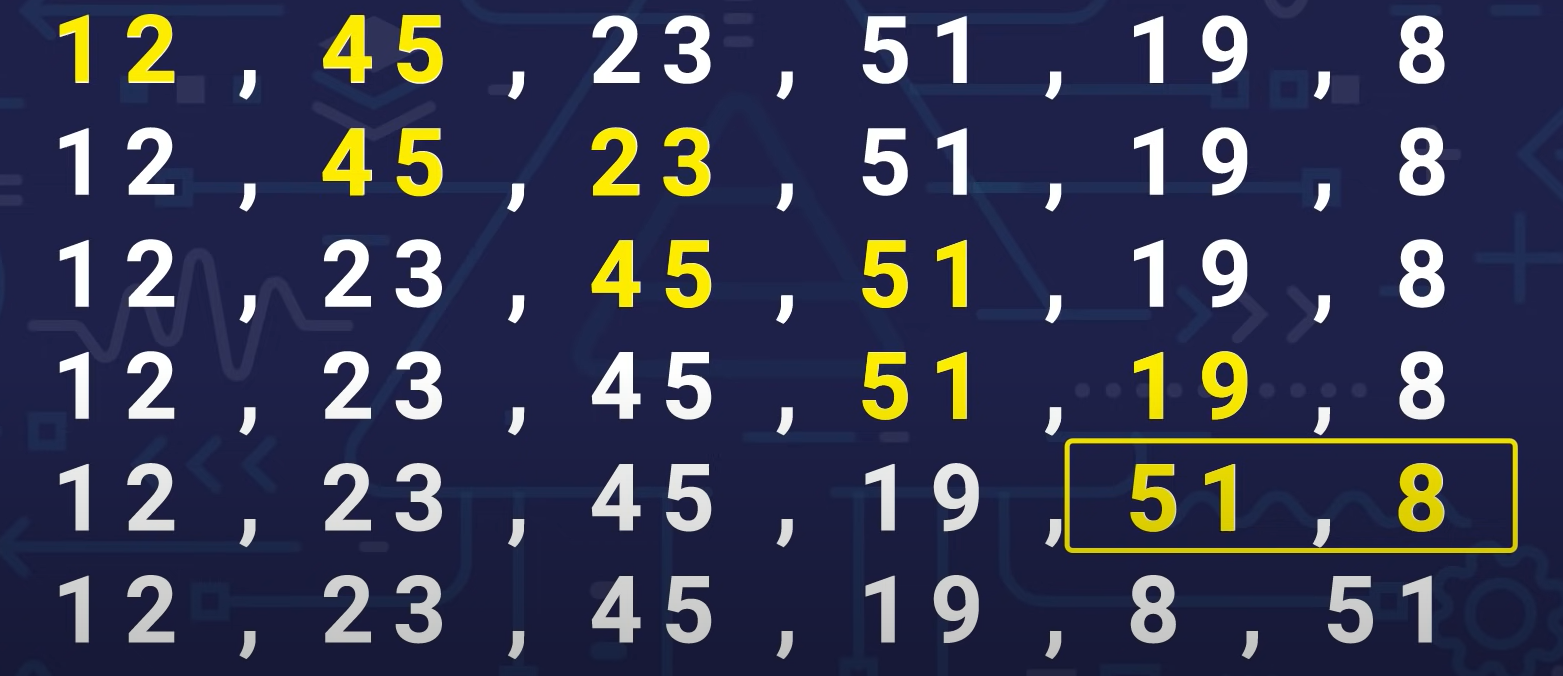
1. SELECTION SORT

Find the minimum element in unsorted array and swap it with the element at beginning.



1. BUBBLE SORT

Repeatedly swap two adjacent elements if they are in wrong order.



1. INSERTION SORT

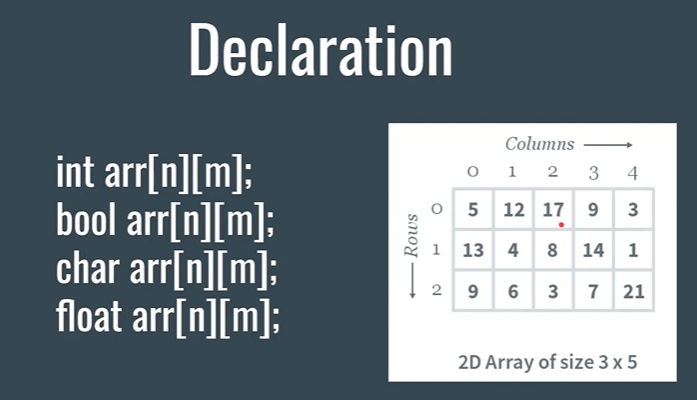
Insert an element from unsorted array to its correct position in sorted array.

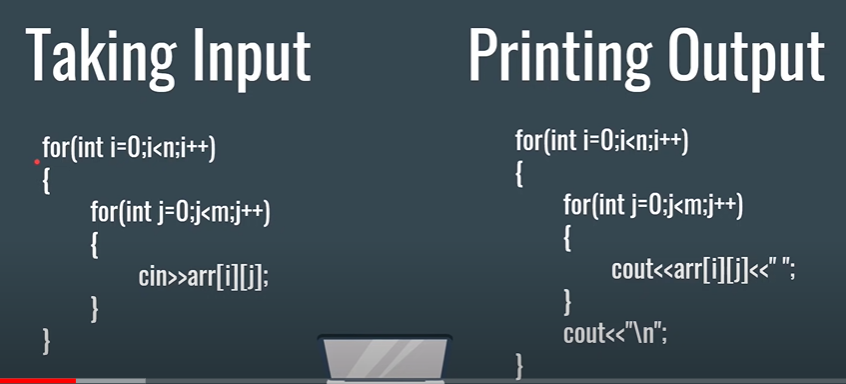
**Subarrays**

A subarray is a contiguous part of an array.

Contiguous means next to each other.

**2-dimensional arrays.**



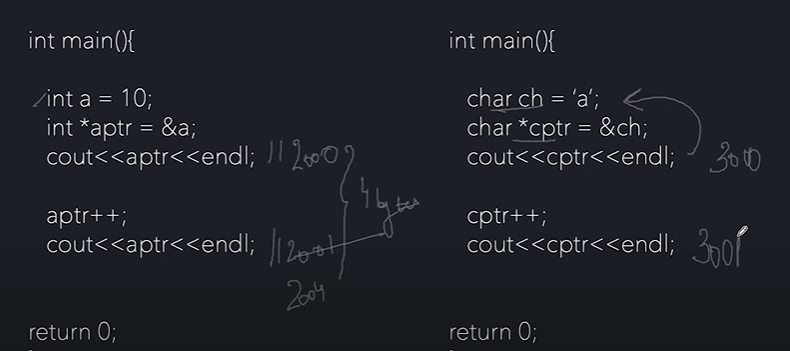


**Character Arrays**

Character arrays are array of characters.

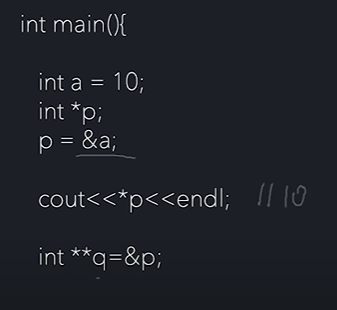
**POINTERS-**

Pointers are variables that store the address of other variables.



This happens because the integer takes 4 bytes whereas the character datatype takes only one byte, hence increasing them will result in these changes shown above.

Pointer to pointer



**Stack**

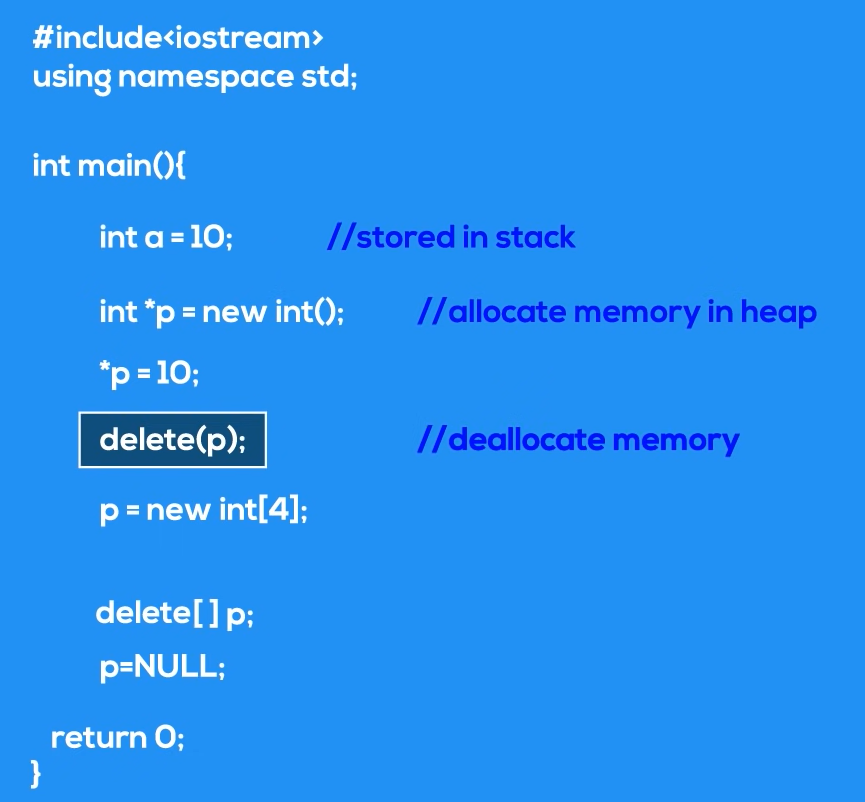
Stack is kind of a storage where all the information regarding the function is been stored, the variables of those function are called as local variables and can’t be accessed by other functions.

And once the function returns anything, it gets popped off from the memory stack.

Since the size of the stack is defined and limited, if we call any function again and again then there comes a point where we have to face the stack overflow situation.

To avoid this situation and get more memory we use the heap, heap is a free energy and is big as our virtual memory of the system.

To use this we have to add the keyword of “new”.



While executing this, a variable will be stored normally in stack, whereas the p will also be stored in stack but will be pointing towards a memory which stores value 10.

Deleting the main value will result in the dangling pointer, which points towards a memory that has nothing stored in it.

Memory leap is a situation where you allocate a memory in heap and don’t delete it. This problem is seen many a times in servers.

**Strings in C++**

There are many advantages of string than using a character array like we need not to mention the size of the string before and we can decide that dynamically.

Concatenate and append functions are easy to use in strings than in character arrays.

Getline function is a pre-defined function in c++, which takes input of string including the spaces.

It is necessary to use it since, the program will not consider the string after the first word i.e. after facing the first space, and hence will only print the first word of the whole string. Getline function thus considers the whole line and prints it without any mistakes.

Functions associated with strings-

String compare function compares two strings and returns the difference between them, if at all the strings are totally equal then the value that our compare function will give will be nothing but 0.

Empty function takes only one string variable at a time and return whether the string is empty or not, we can use this empty function in an if else or if loop.

Erase function in strings erase the desired character/ characters from the string. The arguments in this function are-

1. The number of characters we want to remove from the string.
2. And secondly the position of the character to start removing chars.

Insert function in string inserts the substring, it takes two arguments, and they are-

1. The index/ position that we desire to insert the string into.
2. Secondly, the string that we want to be inserted at that position.

Size function of string returns the size of the whole string. We can also write length instead of size keyword.

Substring function (substr) crops a substring from a full string and saves in a new string variable.

The arguments in the function are-

1. The number of letters we need to crop starting from that position.
2. The position/ index from where we have to start cropping.

Stoi function-

Stoi function in strings converts the string into integers.

Whereas, the to\_string function does the exact opposite thing and converts integers to string.

**BIT Manipulation**

1. Get bit-

This function or operator gives out the bit at any desired position and returns the bit at that particular position.

For this we take bits of 1, i.e. we use left shift operator with the amount of shifts equal to the position, then we compare the given number with the modified bits of integer 1, and return 1 and 0 accordingly.

1. Set bit

Setting bit means changing the value 0 to 1 and some desired position.

Since the bit is been modified, the number will be changed automatically, therefore this function will then return the new number formed with these bits.

1. Clear bit

Clear bit function clears the bit at any given position by the user, clearing here means removing one from that bit and setting it to be zero.

1. Update bit

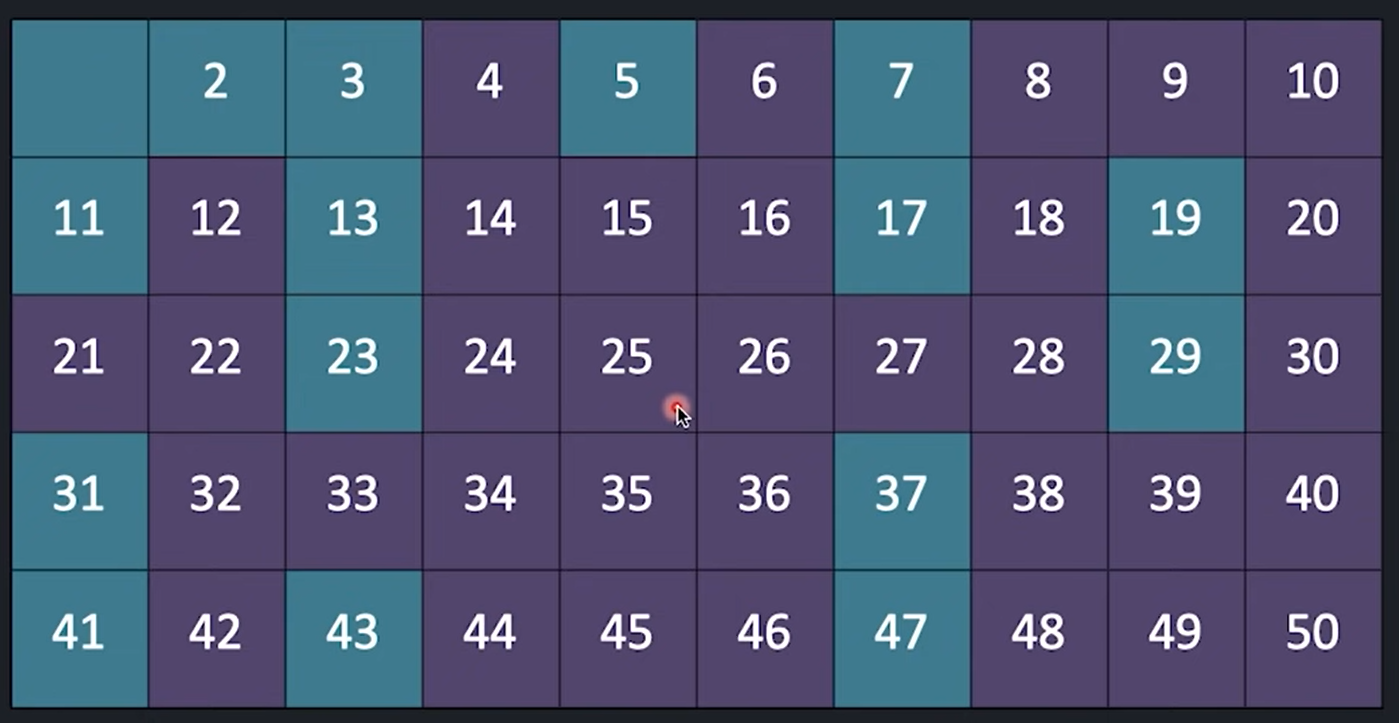
Update bit is a function in bit manipulation that firstly clears the bit and then updates the value to 1.

**Sieve of Eratosthenes**

This is used to get/ print all the prime numbers from a given range of numbers.

In this we start from 2, marking all its multiples, we move ahead with 3 then 4, we actually start with its squares, since the number before its square is already marked as some of the multiples of a previous number before the number we are onto.

We will go till 7 as its square is 49, but we’ll stop at 8 since its square is 64, which is greater than 50 (i.e., greater than our range).



**Inclusion and Exclusion**

Students going for opt 1- n1

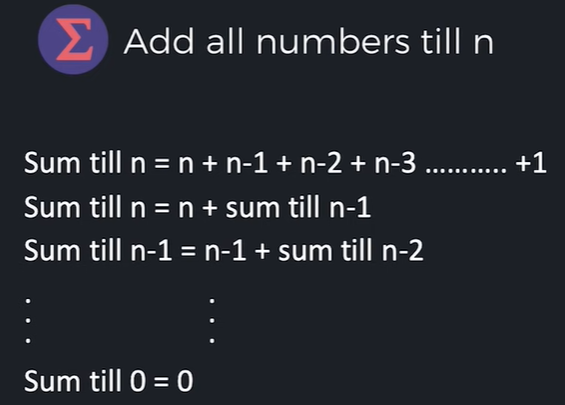
Students going for opt 2- n2

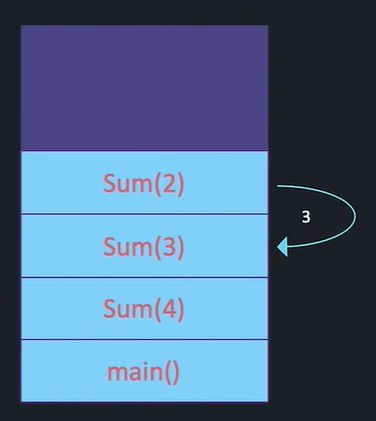
Students opting for both- n3

Therefore, total number of students is equal to => n1 + n2 – n3

**Recursion**

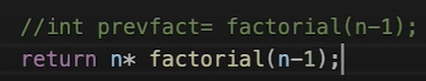
Recursion in simple words, mean a situation when a function calls itself in order to simplify/ shorten the problem and get out of it.





Explanation of recursion example through the stack frame of functions.

A way we can alter the code-



**Backtracking**

Backtracking is basically an algorithm where the program does not run till it reaches the end result, whereas it only runs if the constraints are satisfied, as soon as any one constraint is not satisfied, it will take its way back and try or go towards the next solution.

backtracking is an algorithmic- technique for solving recursive problems by trying to build every problem solution incrementally and removing those solutions that fail to satisfy the constraints of the problems at any point of time.

Rat in a maze problem-

Since we can’t describe a pictorial representation of the maze, we will be showing or describing it through the help of the 2 by 2 matrix of 1 and 0.

**Permutation (return all possible permutation)-**

A permutation is a rearrangement of members of a sequence into a new sequence.

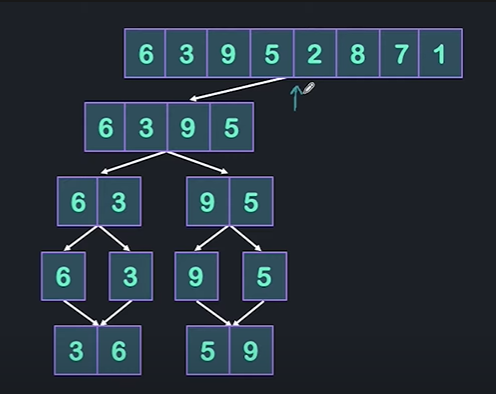
Given an array nums of distinct integers return all the possible permutations. You can return the answer in any order.

N- Queen problem

Chess Board related

**Merge Sort-**

Merge sort technique is based on the divide and conqueror algorithm, in this algorithm we actually divide the main problem into so small problems or sub problems with the help of recursion that the problem becomes easy and can be solved directly.



In the first half we will just divide and divide the arrays into half till we reach a point where we just remain with an array of a single element.

Thereafter we will start coupling them into sorted arrays and further more we will merge them to get a fully sorted array with all elements in it.

**Vectors in C++**

Vectors are like dynamic array where we don’t have to define the size of the array.

Syntax of initiating a vector is to first mention the vector keyword followed by the datatype in the angular braces, lastly the variable name we want to give to the vector of that datatype.

vector<int> v;

It is very necessary for us to include a header file at the beginning named as #include<vectors>.

in order to add elements into the vector we need to use the function called-

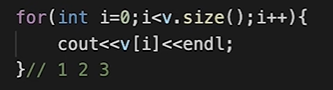
v.push\_back(1);

v.push\_back(2);

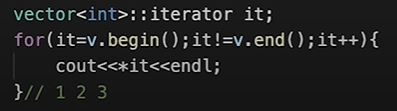
we also have an inbuilt function for vectors that gives the size of the vector called-

v.size();

this can be very helpful while printing all the elements in the vector using a for loop.



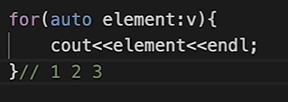
Printing elements using iterator-



We have used \* before the ‘it’ variable since it is a pointer variable.

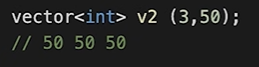
There are also some more functions like v.begin() and v.end() which defines the very first and last element of the vector.

Auto element detects the datatype automatically.



Pop back- pop back function of vector removes the element which is at the last of the vector.

If we know the size of the vector and all the elements are same then we can directly initiate a vector using the following steps-





This will swap all the values between the two vectors.

**Pairs**

Pairs is a type of storage type where we can store a couplets of different datatypes values which might have some relations between them.

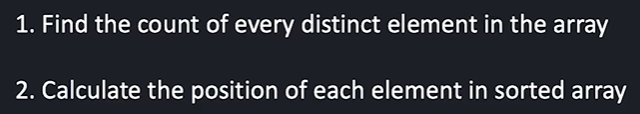
The syntax in order to initiate a pair is as follows:

pair <int ,char> p;

p.first = 3;

p.second = ‘f’;

**count sort**



We will count the number of occurrences of distinct elements present in the array.

