# Let’s solve it

32

# Passing 1d array as an argument

- As array name itself is an address, if provided array to userdefined function, that should be able to access (read / write) memory of array created in the caller function.

- This is useful also in a way, instead of copying and returning so many elements function caller to function called let called f$^n$ access right away.

Syntax

function call: sortudf ( n, arrayname );

function prototype

```
void sortudf( int n, int ptr[ ] );
                OR
void sortudf ( int n, int *ptr );
```

Note that formal argument refering to array address can be written either array notation wise or pointer notation wise. Internally it is pointer only holding base address.

For 1D, the capacity is optional in formal argument.
Because, it is not required for maths anywhere.

---

For 2D, the capacity of ~~smaller~~ ~~inner~~ column dimension is mandatory otherwise array arithmetic for accessing 2d array element is not possible.

i.e. `int arr[ ][5];`

arr[2][3] => $\underline{2 \times 5} + 3$ ✓

Sorting without using udf

```
int main()
{
    int dataarr[]={50,40,30,20,10};
    int n=5;

    print before

        sort        for
                        for
                            if
                                exchange

    print after
}
```

Starting with UDF

```
void sortudf( int n, int dataarr[ ] );
int main()
{    int data[ ]={50,40,30,20,10}; int n;
     print before
      sortudf (n, dataarr);
     print after
}
void sortudf(int n, int dataarr[ ])
{        int temp;
           for: for  if exchange
   }
```

Note that the dataarray is

printed ascending sorted that

proves that array is by default

call by address.

Called f^n has access to memory

of caller.

## 2d array

sorting table of strings

no. of words — word size

```
void sortwdfts( int n, char dataarr[ ][5] );

int main( )
{
    char dataarr[ ][5] = { " ", " ", " ", " ", " " };
    int n=5;
// print before
    sortwdfts( n, dataarr );
// ~~print~~ sorted
}
void sortwdfts( int n, char dataarr[ ][5] )
{
    char temp[5];
        for for if exchange
}
```

Note the syntax
in prototype and declaration,
only the first left most capacity/size
is optional, all others must be
specified. i.e. 51 number of
letters/columns.