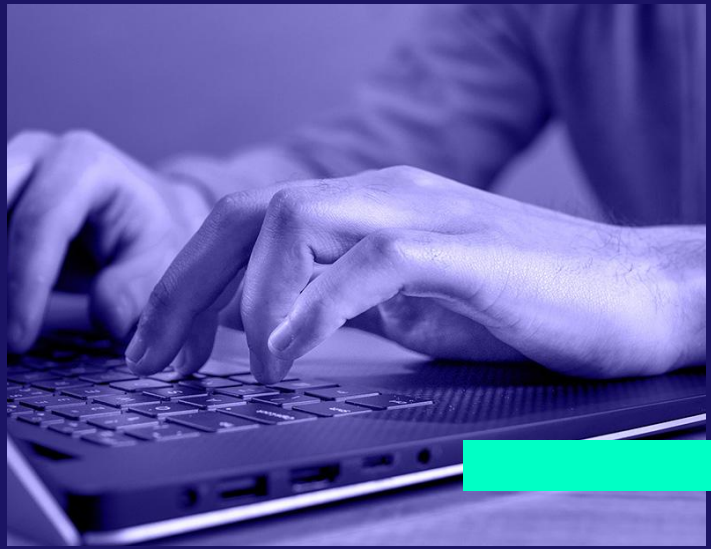
C
Intermediate
LESSON 1

TABLE OF CONTENTS

INTRODUCTION

Software Engineer != Programmer

1. We need to make sure whether this program will work with program design or not and more.
2. We must satisfy our customer needs.



RULES

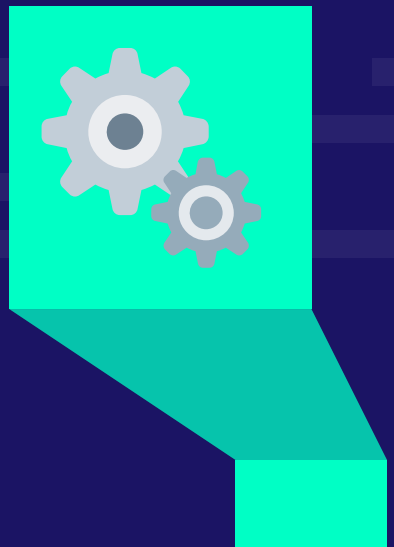
1. Readable code
2. Appropriate variable name
3. Type of code

3 THINGS TO LOOK FOR

Speed

Competitive programming

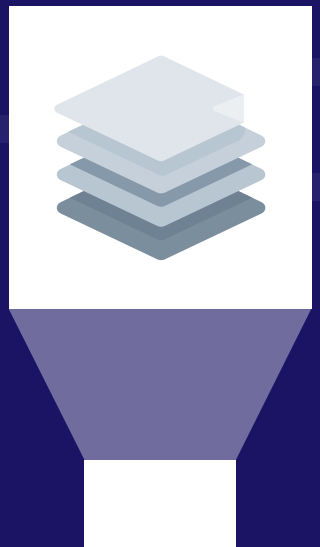
Server Communication



Maintenance

Big project

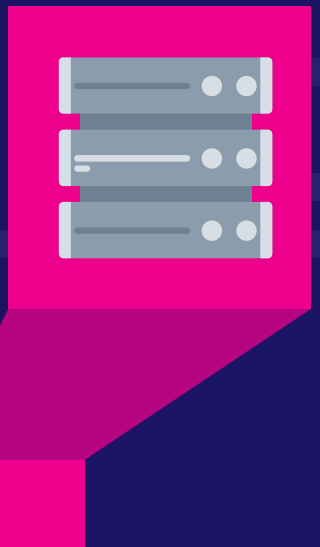
Working with others



Memory

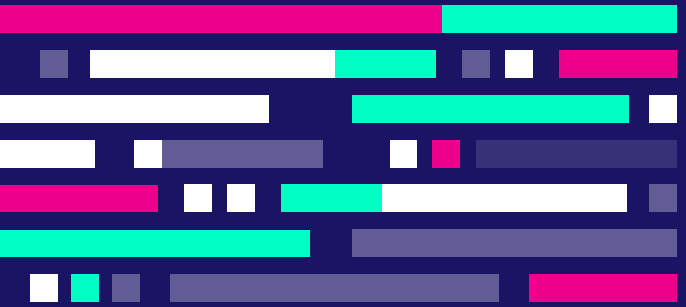
Limited memory of

each device



01

ARRAY



What can it do?

Array is use for storing many values with the same type in one variable name.

SYNTAX

(data type) (arr name)[(size)]

Example:

int array[5];

double array [10];

</>

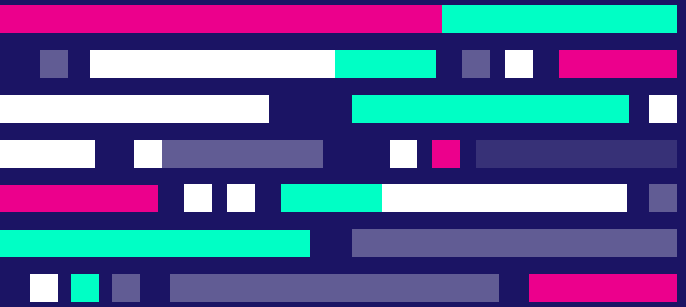
Input Array

```
scanf(“%d”, &arr[(index)]);
scanf(“%lf”,&arr[(index)]);
```

Output Array

```
printf(“%d”, arr[(index)]);
```

You cannot display all the value by using one command like Python!



Why is it important?

Let's look at an example question:
Q: Create a program that first ask user how many integers user want to enter then after the user enters all integers then the program must display all value that the user had enter.

Note: Maximum 100 integers

Without Array

```
int val1, val2,...,val100;
int amount;
scanf("%d", &amount);
scanf("%c");
for(int i=0; i<amount; i++){
      if(i==0)
            scanf("%d", &val1);
      else if(i==1)
            scanf("%d", &val2);
      …
      else if(i==99)
            scanf("%d", &val100);
}
for(int i=0; i<amount; i++){
      if(i==0)
            printf("%d", val1);
      else if(i==1)
            printf("%d", val2);
      …
      else if(i==99)
            printf("%d", val100);
}
```

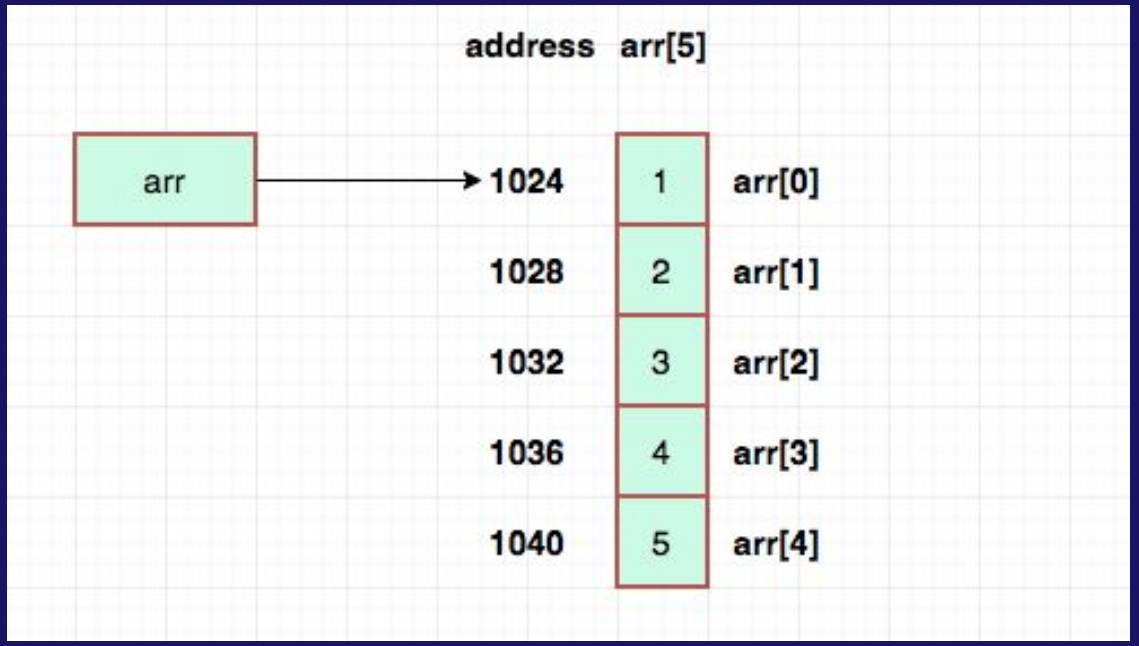
Not efficient af
Not dynamic
Bad code
Estimated line: 408 lines

With an Array

```
int usin;
scanf("%d", &usin);
int arr[usin];
for(int i=0; i < usin; i++){
    scanf("%d",&arr[i]);
}
for(int i=0; i < usin; i++){
    printf("%d\n",arr[i]);
}
```

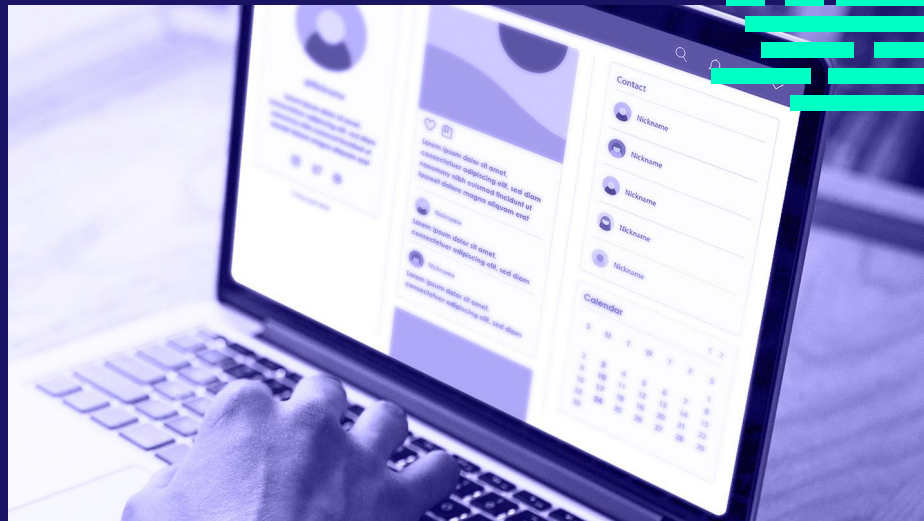
Very efficient
A little dynamic
Good code
9 Lines of code

How does it works?



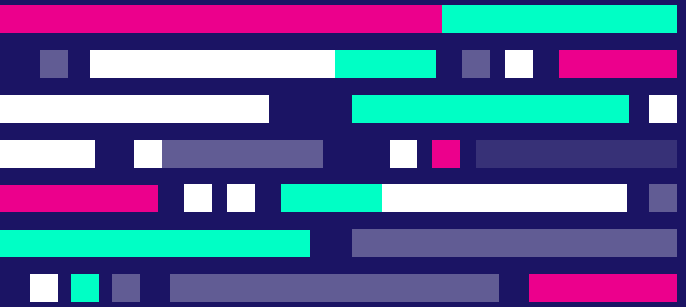
ERROR Cases

1. Index out of range
2. Changing the size of array
3. Type does not match



02

STRING



What is it?

A char array, that's it XD

SYNTAX

```
char (string name)[(size)];
char *(string name);
```

Example:

```
char str[5] = “Hello”;
char str[] = “Kris”;
char *str = “Test”;
```

Input String

```
scanf(“%s”, str);
gets(str);
```

Output String

```
printf(“%s”, str);
puts(str);
```

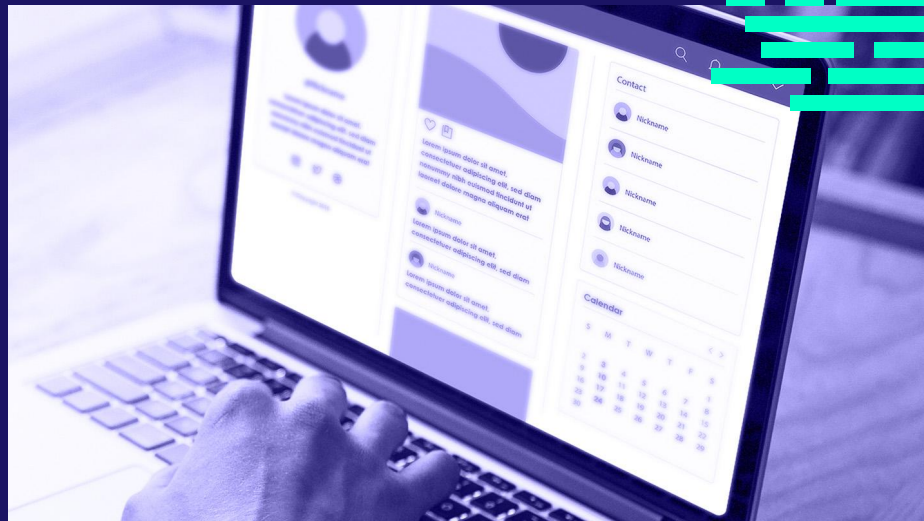
How does it works?

```
char name[6] = { 'T', 'O', 'M', 'M', 'Y', '\0' };
char name[ ]  = "TOMMY"
```

T	O	M	M	Y	\0
65120	65121	65122	65123	65124	65125

ERROR Cases

1. Most of the errors you can get from a normal array
2. Use ‘ ‘ for string will cause error
3. Use “ “ for char will cause error
4. Forget about ‘\0’



03

STRING MANIPULATION

LIBRARY

string.h

```
#include <string.h>
```

COMMANDS

strlen() - computes string's length
strcpy() - copies a string to another
strcat() - joins two strings
strcmp() - compares two strings

Example

```c
#include <stdio.h>
#include <string.h>
#define BUFSIZE 300

int main()
{
    char str[BUFSIZE] , temp[BUFSIZE];
    scanf("%s", str);
    printf("%d ", strcmp(str,temp));
    strcpy(temp,str);
    printf("%d ", strcmp(str,temp));
    strcat(temp, " World");
    printf("%d ", strcmp(str,temp));
}
```

Input:
Hello
Output:
1 0 -1

```
strcmp == 0: same string
strcmp == 1: if dest has more string than source
strcmp == -1: if dest has lesser string than source
```

04

LAB


slidesgo