Build a Tic Tac Toe in C++

Tic Tac Toe Game

→ 2 player game → Player mechanism

→ Board → Board mechanism

→ Win | Lose | Draw → Result mechanism

→ Game flow. → input taking mechanism

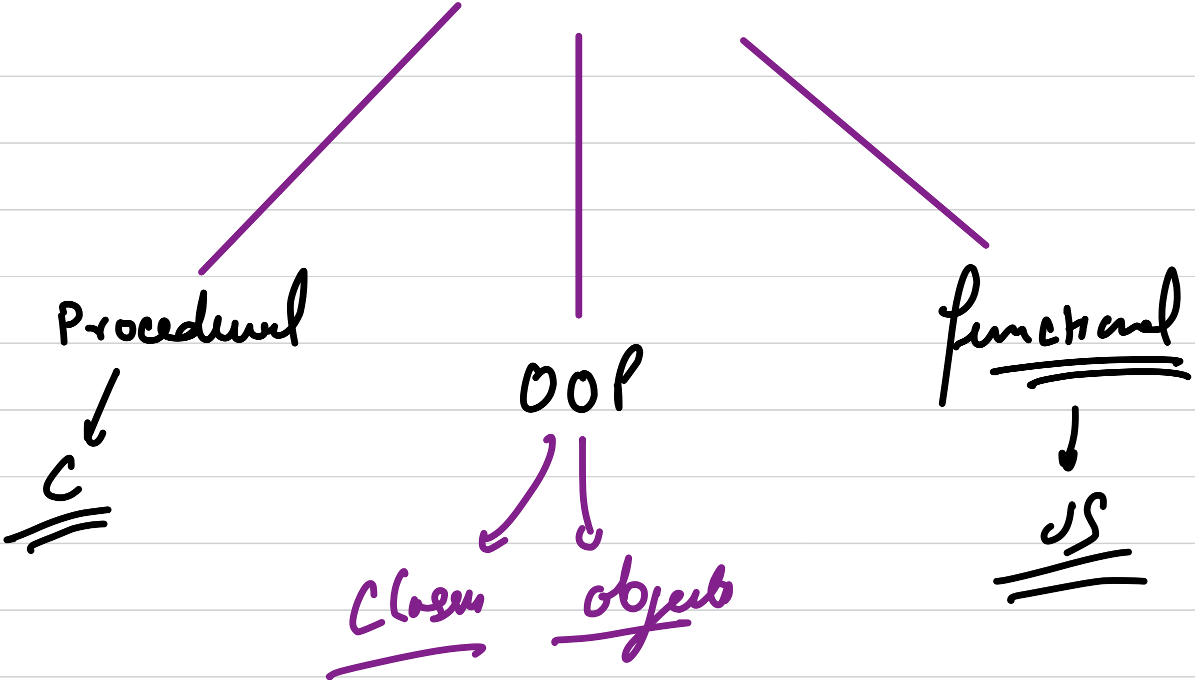
→ Logging mechanism

Programming

Language

CPU

Cpp, java, ruby, python etc.


Procedural
C
OOP
Classes
objects
functional
JS

CPP

Why CPP? → **Powerful** → **Rich STL**

↓

PBDS

↳ It interacts directly with **hardware**

↳ It gives us manual access to manage **memory**.

↳ It **doesn't** **abstracts**

↳ faster than java / Python / **ruby**

↳ **DS / Algo**

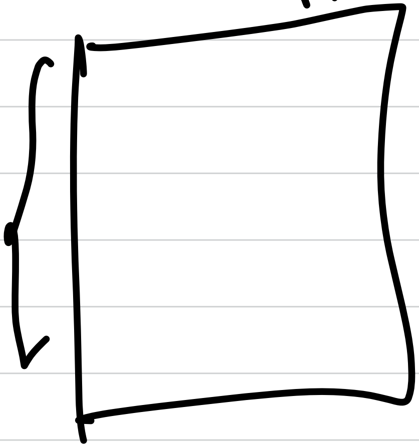
code → cpp program

run

process

extension

.cpp



```
#include <iostream>
```

preprocessor **directive**

instruction to include file

filename

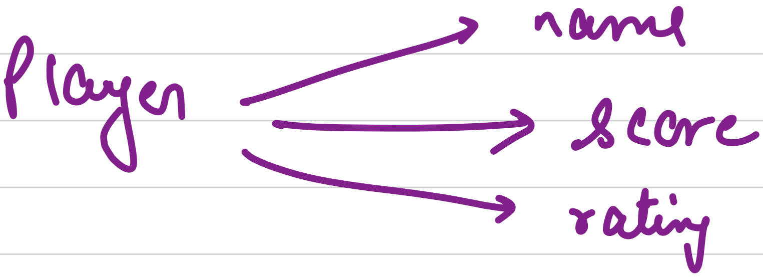
keywords ←

;] → one instruction **completed**

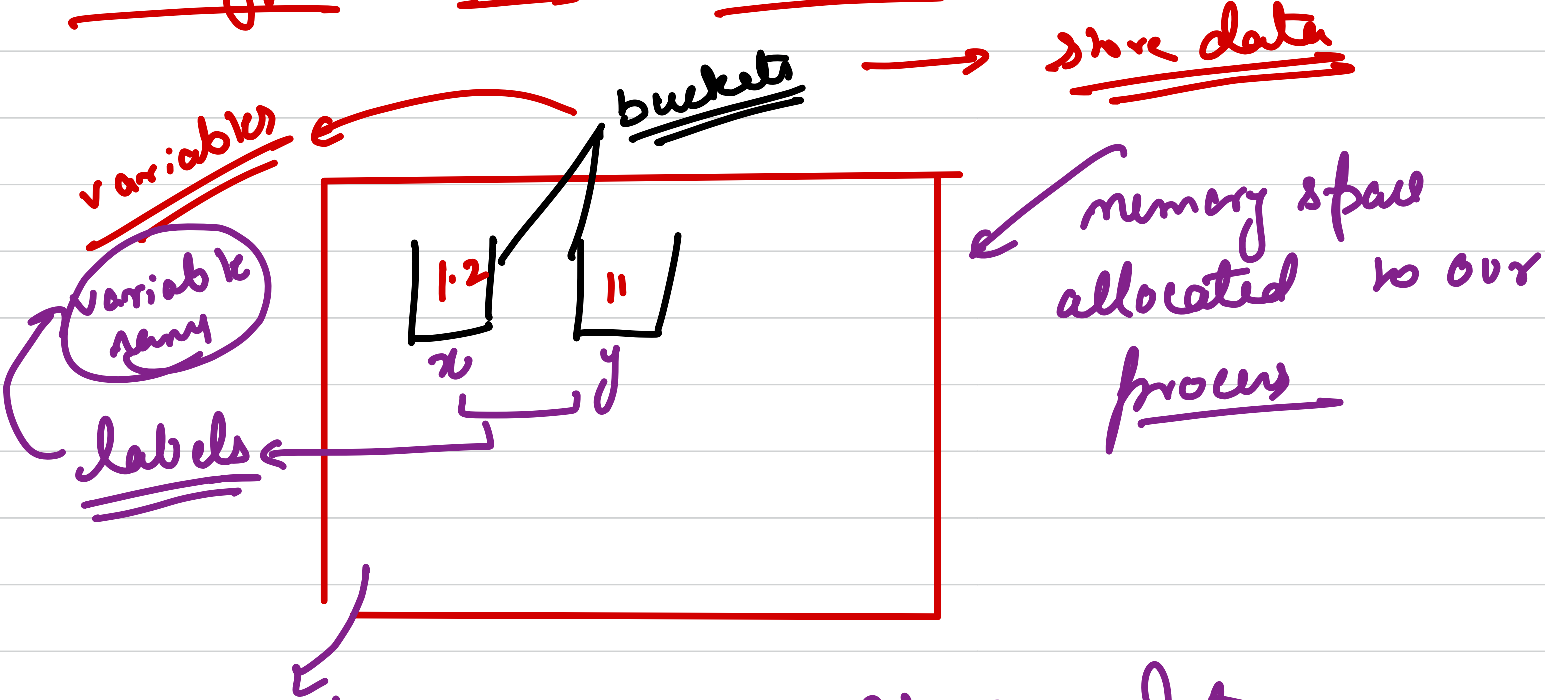
text → $"\underbrace{\qquad\qquad\qquad\qquad}_{\textbf{text}}"$

Building the player mechanism.

to show any entity, we need to make them possess multiple different type of value.



Datatypes and Variables



use the memory to store data relevant to our process.

```
<type> <name> = <value> ;
```

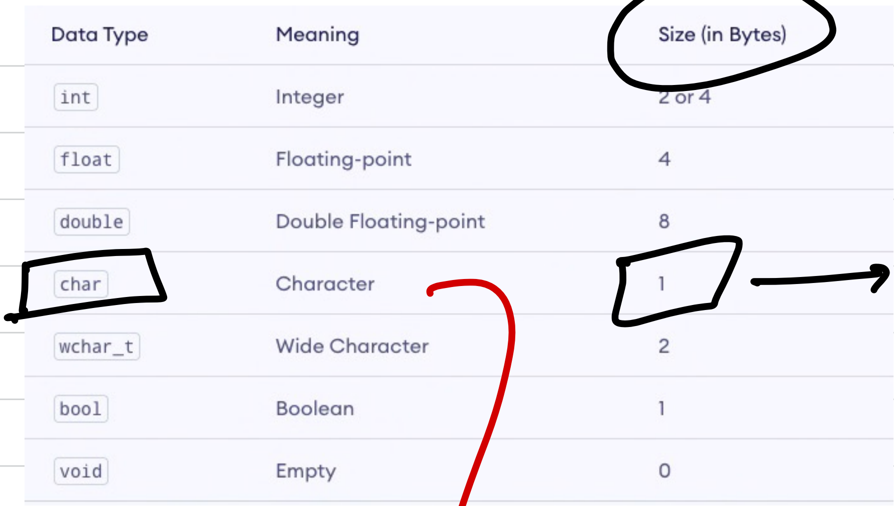
datatype

label of bucket

not mandatory

Ex →

```
int x = 12;
float y;
```



Data Type	Meaning	Size (in Bytes)
int	Integer	2 or 4
float	Floating-point	4
double	Double Floating-point	8
char	Character	1
wchar_t	Wide Character	2
bool	Boolean	1
void	Empty	0

Compiler and System dependent

2^{32}

1 byte → 8 bits

total 256

↳ → byte value 0 – 255

long

long long int

short

byte

etc

ascii nos

14 →

$$\frac{2}{0/1}\ \frac{2}{0/1}\ \frac{2}{0/1}\ \frac{2}{0/1}\ \frac{2}{0/1}\ \frac{2}{0/1}\ \frac{2}{0/1}\ \frac{2}{0/1}$$

2^8 → 256 chars

int → +ve

→ −ve

→ 0

for a 4byte int → min → −2147483648

max → 214783647

0 → +ve

1 → -ve

$\boxed{0/1}\ \underbrace{0\ 0\ 0\ 0\ 0\ 0\ 0}_{7}$ → +0 -0

MSB sign bit

-ve are stored in 2's complement form.

$$
\begin{array}{r}
1011 \\
\text{1's} \hookrightarrow 0100 \\
+1 \\
\hline
0101
\end{array}
$$

2 → 10

1's → 101

$$\begin{array}{r} +1 \end{array}$$

2's → 10

$$\begin{array}{cccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
 & & & & & & 1 & \\
1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\
 & & & & & & + & 1 \\
\hline
1 & 1 & 1 & 1 & 1 & 1 & 1 & 0
\end{array}$$

2's complement

8bit

— — — — — ... — — — —   — —

↳ 32 bits

$\boxed{0}$ 0 0 0 0 1 0 1

— — — — — — — —

↳ 8 bits

1 1 1 0 0 1 0 1

$2^{8-1}-1$

↳ $2^{7}-1$

$2^{7} + 2^{6} + 2^{5} + 2^{2} + 2^{0}$

⇒ 128 + 64 32 4 1

special chars → '\n' → newline

'\t' → tab

' ' → space

Chars → single quote

text → " " → double quote

struct → structure