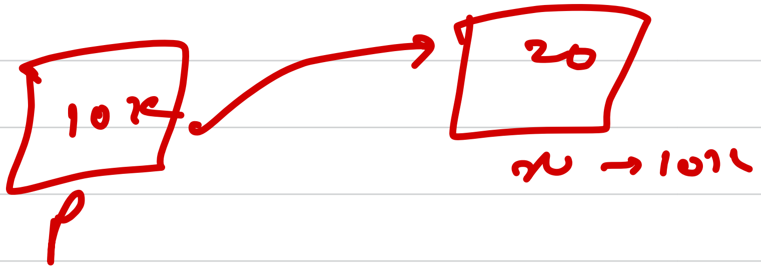
Types Of Pointers :

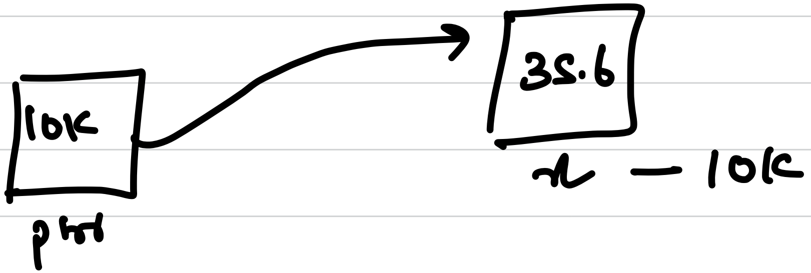
1) **Wild Pointer** → **Uninitialized Pointers** . Because of that they can have any random value, inside them that can later cause **problems**

```
int *p;
```
← wild pointer

2) Dangling Pointer → A pointer pointing to a memory location that was previously available but now has been deallocated.



3) Void Pointer →



float* ptr = &x;

$\underbrace{\text{float}}$ → what type of bucket it is pointing to.

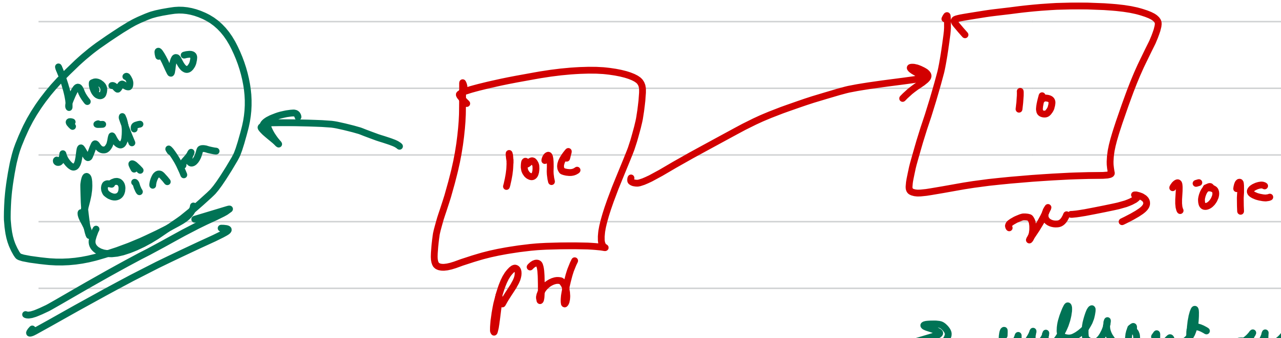
void * ptr $\longrightarrow$ $\boxed{\text{malloc}}$

It doesn't have a specific type associated.
The type of data can be anything.

↳ They cannot be dereferenced

Dereferencing

int *ptr = &x;

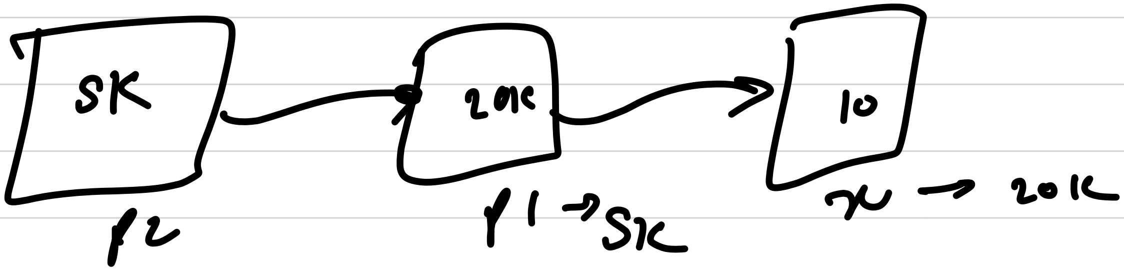


int y = *ptr

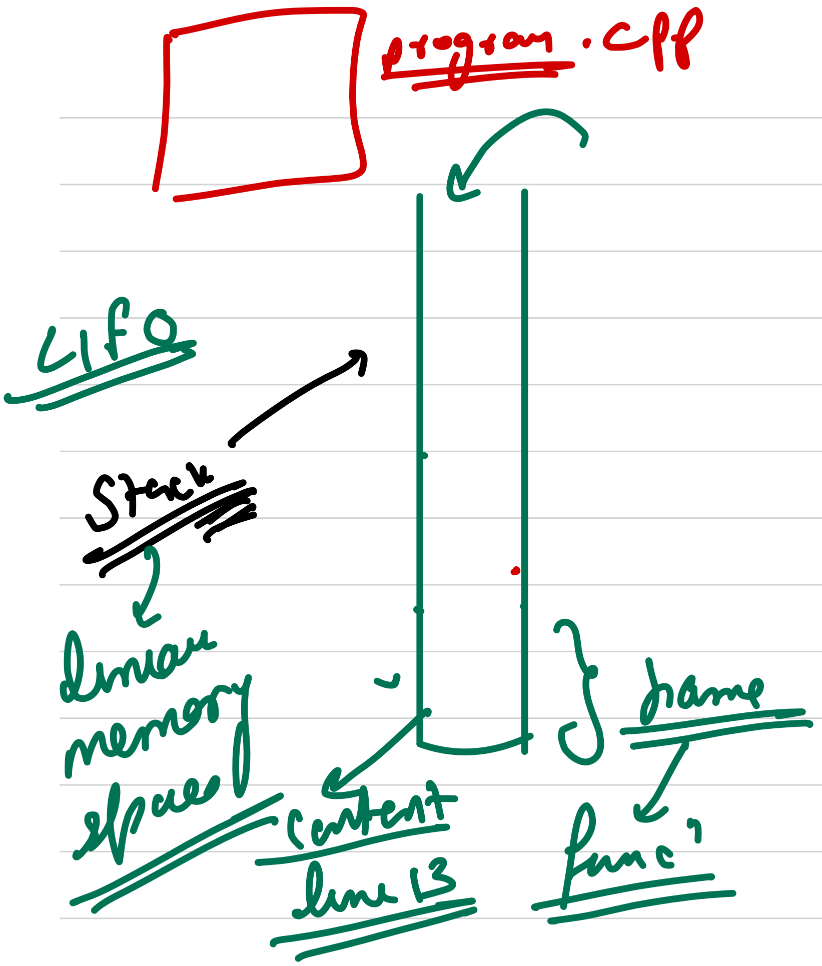
without value → dereferencing operator

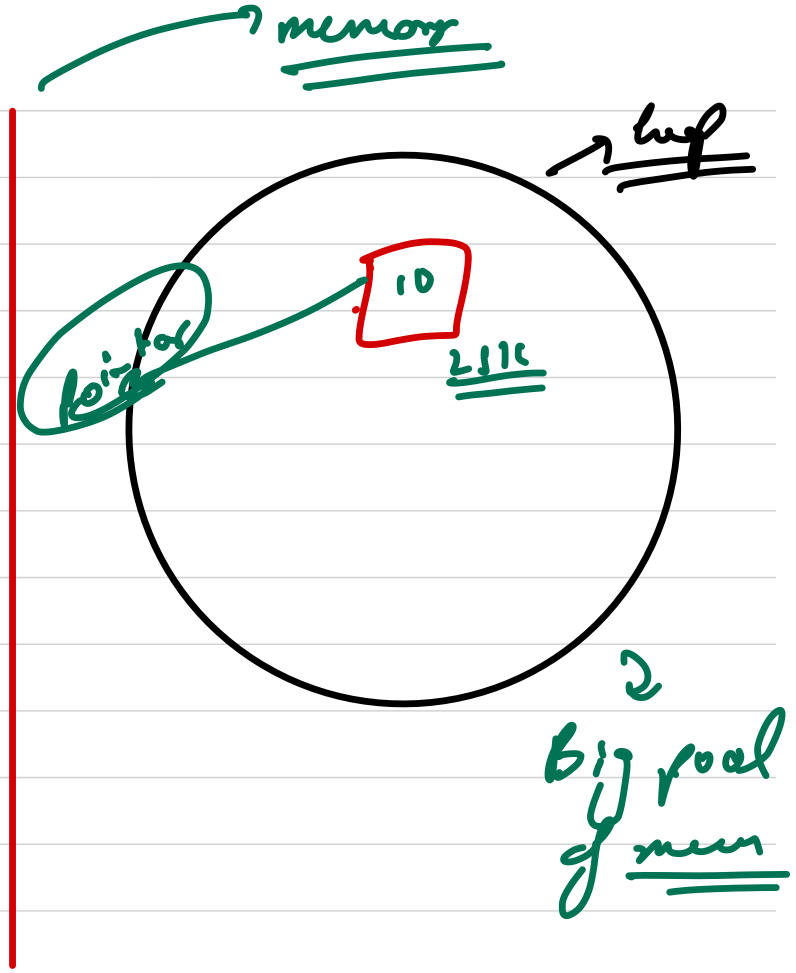
cout << *ptr << "\n";

```
int x=10;
int *p1 = &x;
int** p2 = &p1;
```



$**p2 \rightarrow \underline{\underline{10}}$


program.cpp
LIFO
Stack
linear memory space
content line 13
frame
func^n


memory
heap
pointer
10
2116
Big pool of mem

whenever we call a funcⁿ

$$\underline{\underline{\text{main()}}} \longrightarrow \text{fun()} \longrightarrow \text{gun()} \; \ldots$$

```
#include <iostream>

void fun() {
std::cout << "fun" << "\n";
}

int main() {

  → int x = 10;
  → fun(); ←
    return 0;
}
```

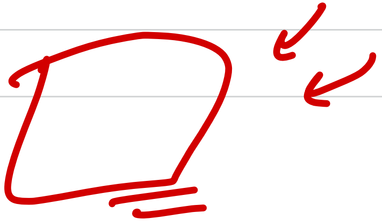
↳ function → malloc (size) → 4 4byte

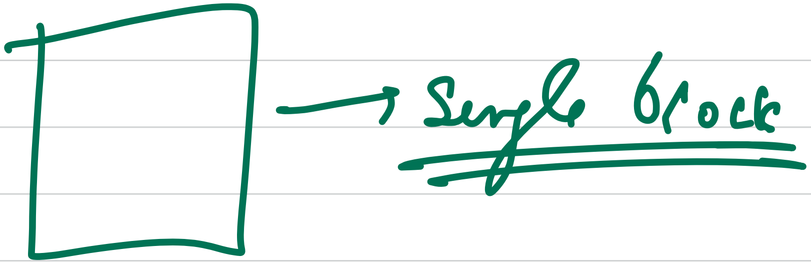
void *

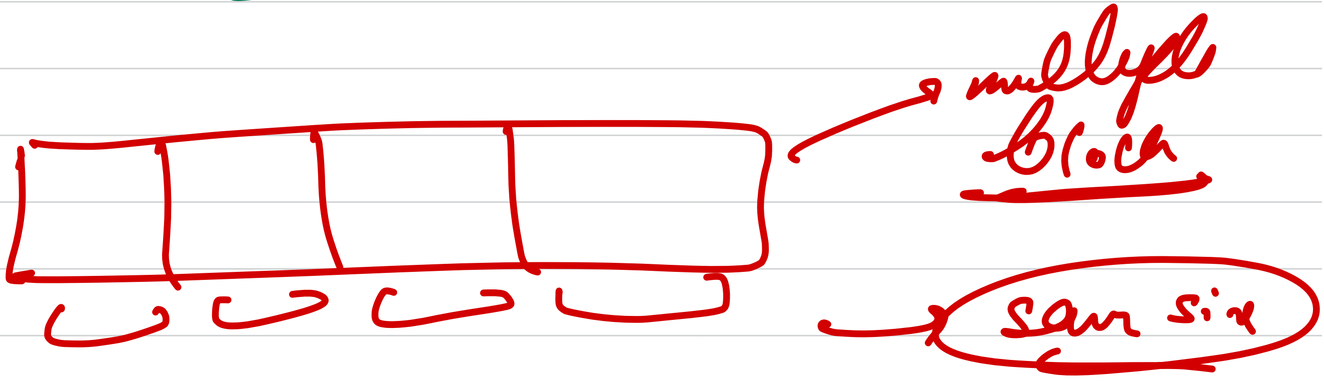
Pointer
1GB

typecast

(int *)





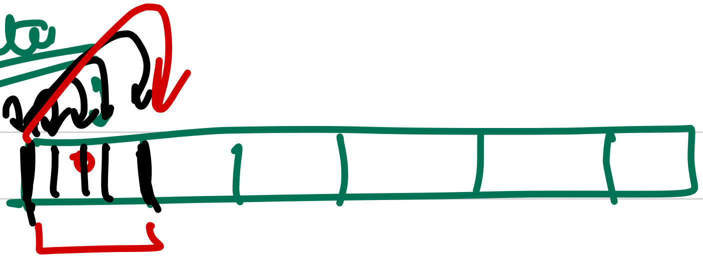


calloc (4 , 5)

no of block — size of bucket

arr → pointer

it stores address of 0 arr / 0th index



int arr[10];

label

pointer arithmetic

integer → 4 byte

constant arr

arr + 1

Smart Pointer

Smart Pointers are an abstract interface to actual raw pointers but with additional benefit of auto memory management.

$$\begin{tikzcd}
 & \text{unique ptr} \\
3) \arrow[ur] \arrow[r] \arrow[dr] & \text{shared} \\
 & \text{weak}
\end{tikzcd}$$

Ref count