LECTURE - 32

POINTERS (PART 02) AND ARRAYS

PROGRAMMING IN 'C'

NULL Pointer :

Nothing

Nothing → NULL or NaN or None

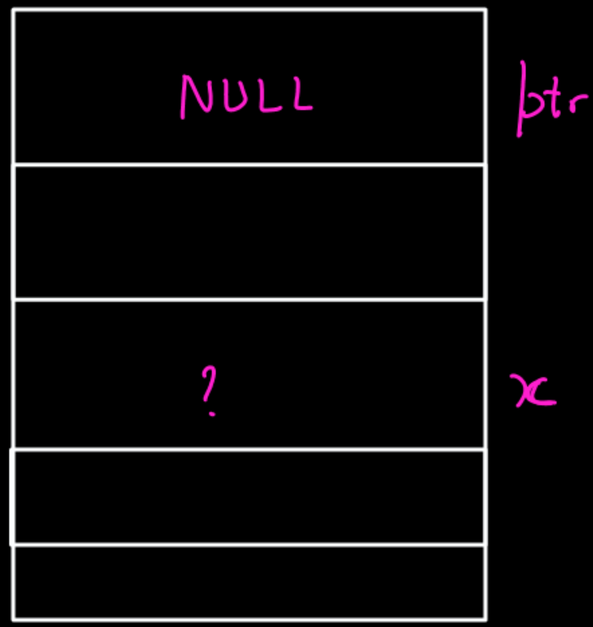
```
int main() {
    int *ptr = NULL;
    printf("%d", ptr);

    int *x;
    printf("%d", x);

    return 0;
}
```

ptr = NULL

printf("%d", ptr); ⇒ Print → ptr → NULL

int *x; → declare

printf("%d", x); → garbage Value or Unwanted behaviour or 0 or Warning



Why NULL Pointer (Void Pointer)?

two Major Cases

(A) To Prevent from Garbage Value ← Because a random garbage value may be the address of other Variable.

(B) To Prevent from Dangling Pointers :

↓

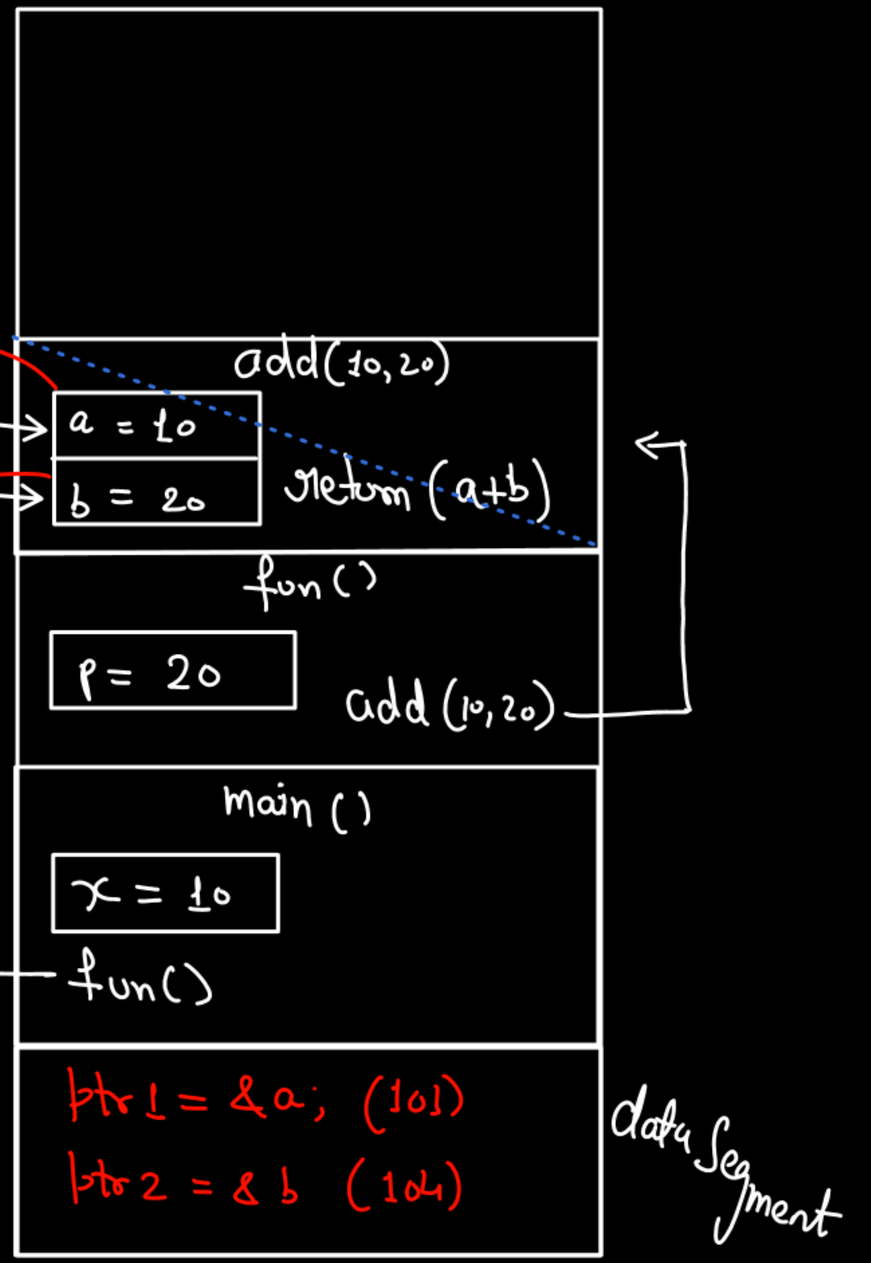
खोजाना ← भटका हुआ

Dangling Pointers

Static

pointer → ptr1 = &a; (101)

ptr2 = &b; (104)



After Some time,

[a & b destroyed]

then,

What the pointer ptr1 & ptr2 points?

Dangling Pointer

The pointer who holds the address of non existing variable, or the destroyed variable are termed as dangling pointer

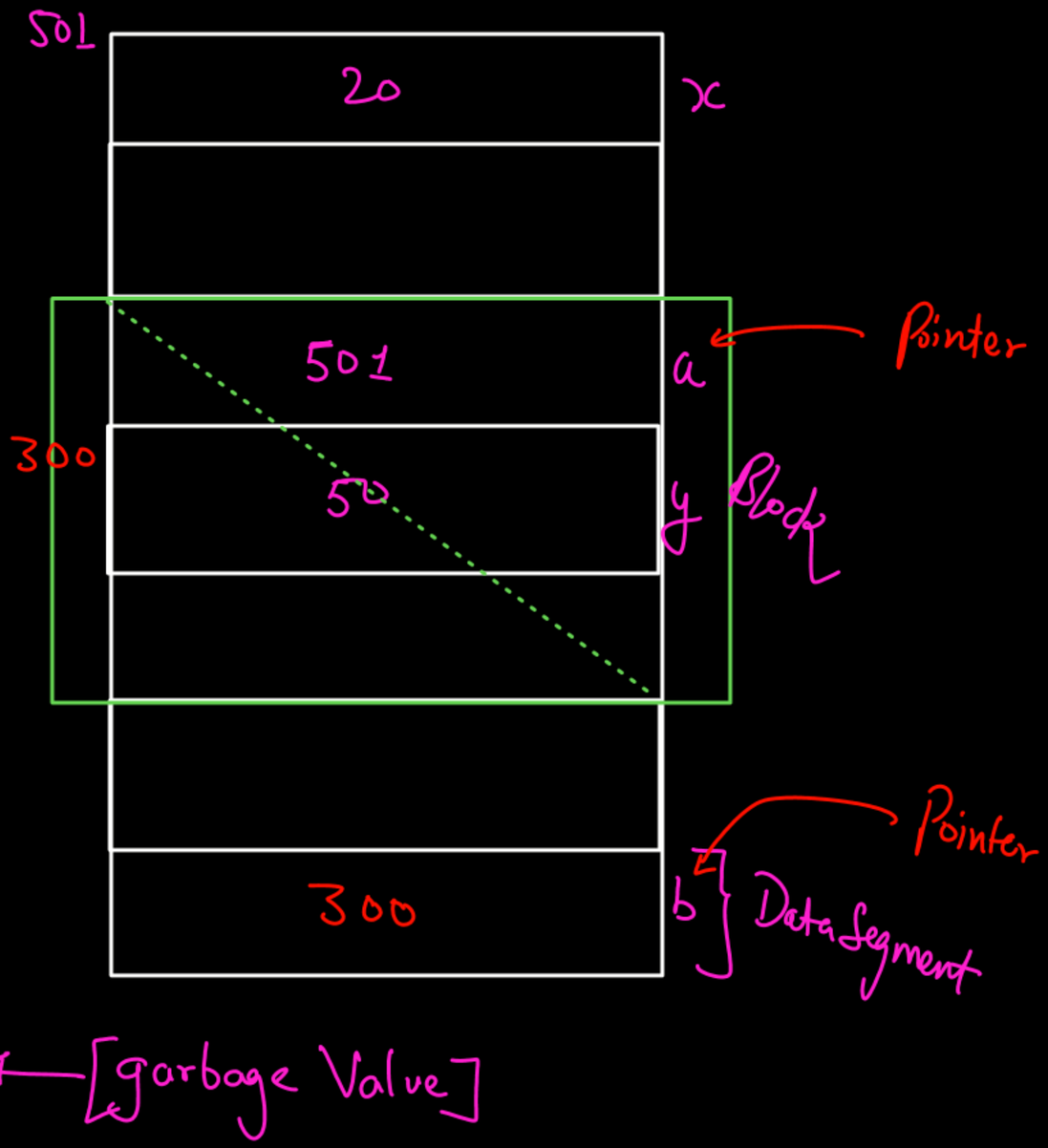
```
int main ( ) {
    int x = 20;
    {
      int * a = & x ;
      int y= 50 ;
      static int * b= &y ;
    }
    printf ("%d", b );
    printf ("%d", *b);
    return 0;
}
```

300

dangling Pointer

Value at (b)

Value at (300) ← [garbage Value]



Dangling Pointers → Dangerous

To Prevent from dangling Pointers

↳ We use NULL Pointers, or set the pointer variable to NULL Before destroying a referenced variable.

or

free the pointer Variable

ज्ञान से ज्ञान

⇒ Minimal assignment of the address of local variable will keep program safe from dangling pointer.

↓

Scope & duration

↓

within Block

Arrays in C

→ Collection of Homogeneous elements in Successive memory location.
is called an array. ← Data Structure

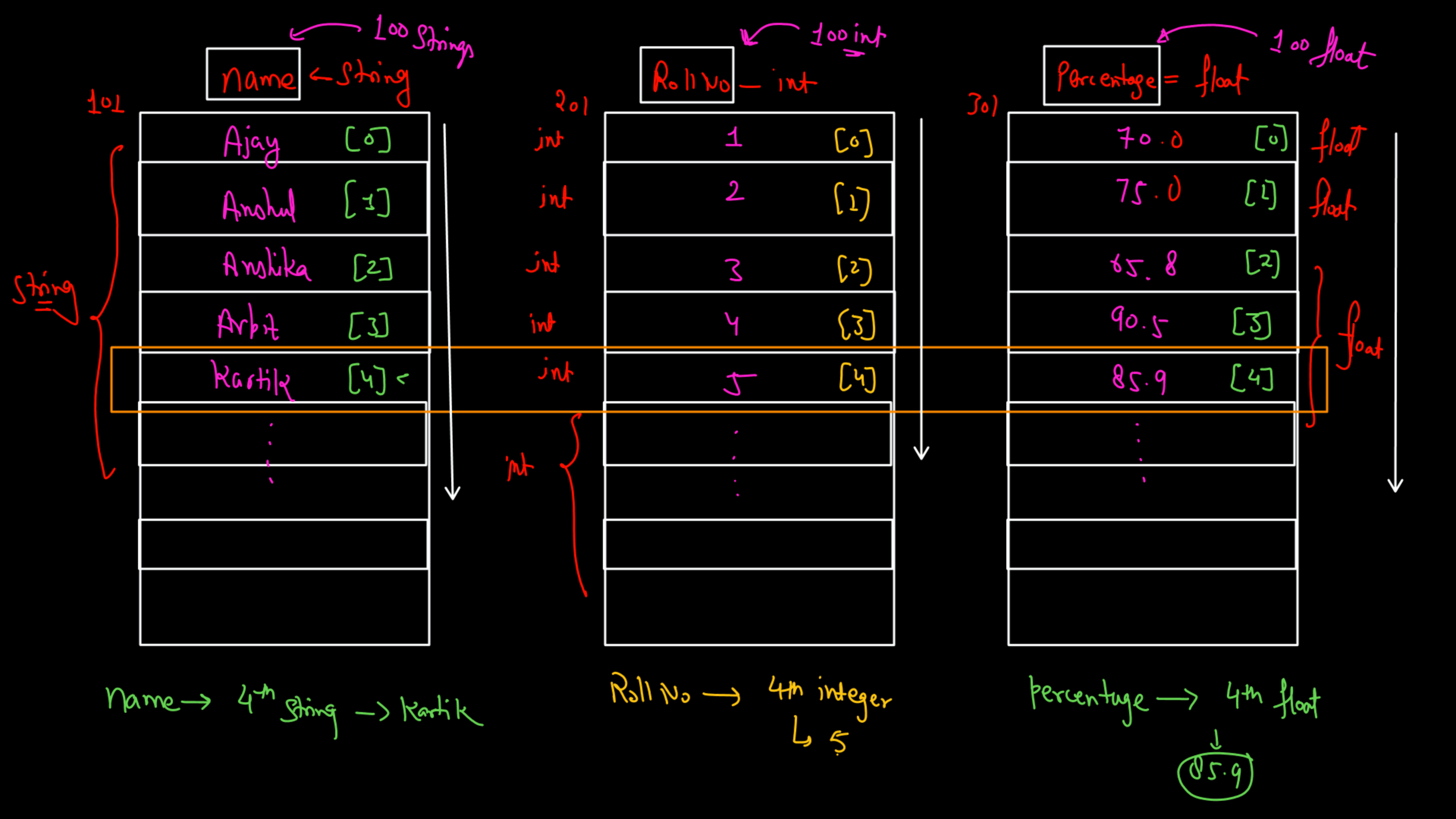
Suppose, Students = 100 (Name, RollNo, perc)

String
Name = | "Ajay" | "Anshul" | "Anshika" | "Arpit" | "Kartik" | ....

int
RollNo = | 1 | 2 | 3 | 4 | 5 | ...

float
Percentage = | 70.0 | 75.0 | 65.8 | 90.5 | 85.9 | .....

Name → String
RollNo → int → 4 Bytes
Percentage → float → 4 Bytes


100 Strings
name ← String
101
Ajay [0]
Anshul [1]
Anshika [2]
Arbit [3]
Kartik [4]
String
100 int
Roll No – int
201
1 [0]
2 [1]
3 [2]
4 [3]
5 [4]
int
100 float
Percentage = float
301
70.0 [0]
75.0 [1]
85.8 [2]
90.5 [3]
85.9 [4]
float
name → 4th string → Kartik
Roll No → 4th integer
5
Percentage → 4th float
85.9

Roll No → 100 int

1 int = 4 Bytes

Size of (Roll No) → [100 int x 4]

⇒ 400 Bytes

Here, Roll No is an array !

⇓

array of integer

Percentage → 100 floats

1 float = 4 Bytes

Size of (Percentage)

↓

100 x 4 = 400 Bytes

Array of float

* Storing multiple values in a single Variable

Array → Data Structure