CS & IT ENGINEERING

Programming in C

String in C Programming

Lec- 02

By- Pankaj Sharma sir

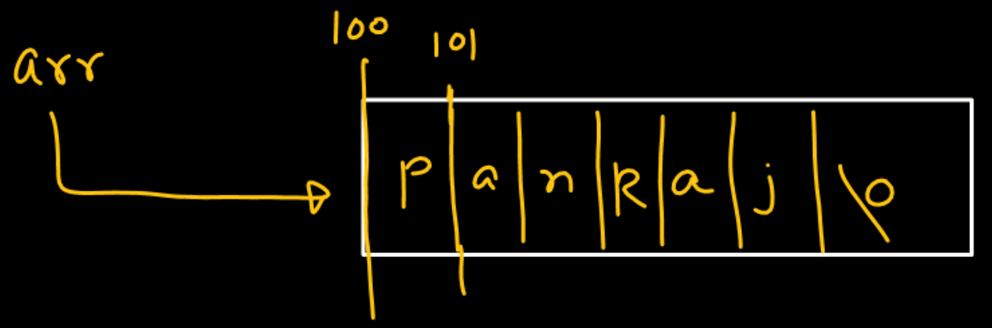
Strings-2

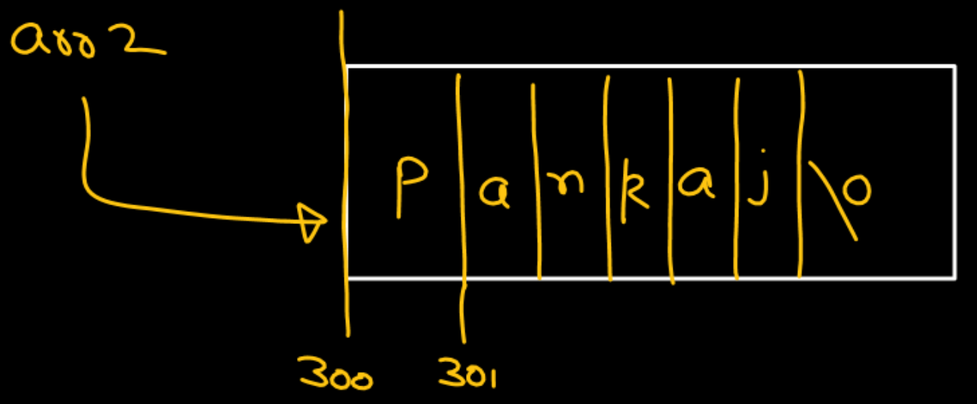
1.

```
char arr[ ] = "Pankaj" ;

char arr2[ ] = "Pankaj" ;

if ((arr + 1) == (arr2 + 1))
        bf("1");
else
        bf("0");
```





O/P : 0

2.

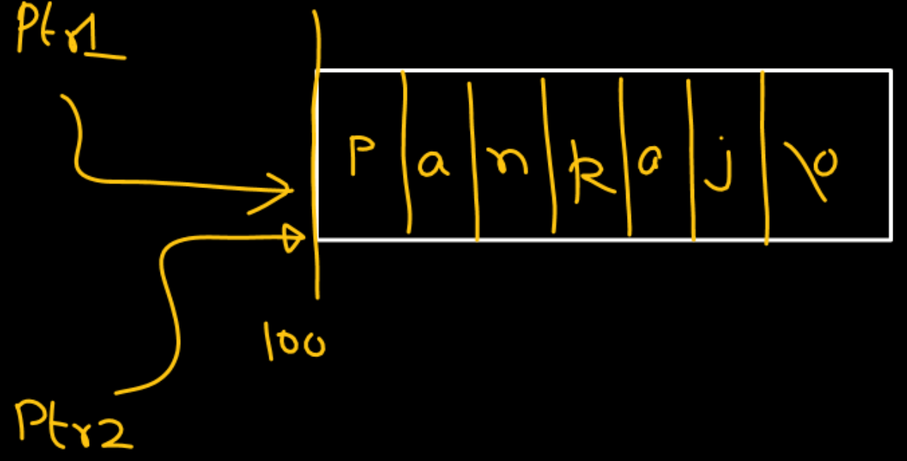
```
char *Ptr1 = "Pankaj";
char *Ptr2 = "Pankaj";
```

100

100

Read only Area

(No one can change it)

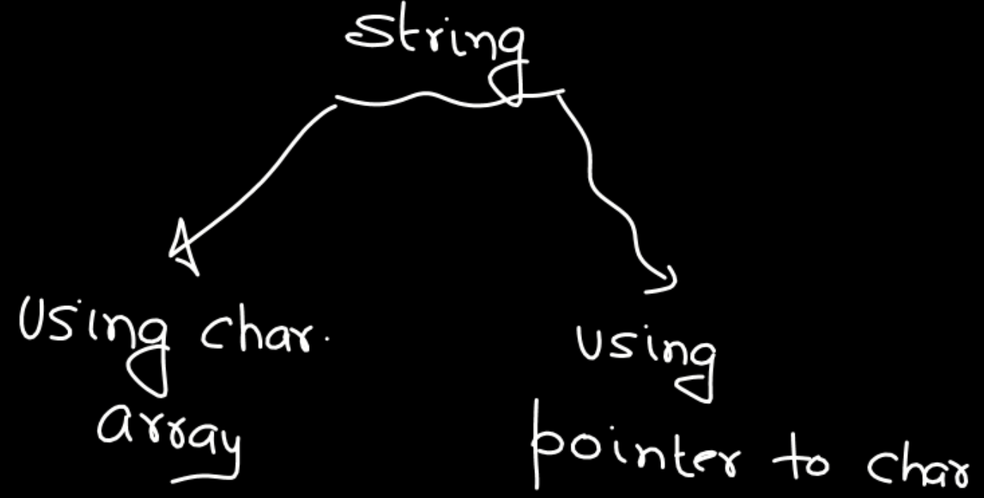



```
if (Ptr1 == Ptr2)
    bf("1");
else
    bf("2");
```

2.

```
if ("Gate"[1] == "Gate Wallah"[6])
    pf("Maza agya");
else
    pf("Sabko");
```

'a' 'a'



1 string — 1 char. array
2 string — 2 char array
3 string — 3 char array

```
char arr[10] = "Pankaj";
```
1-D array of char

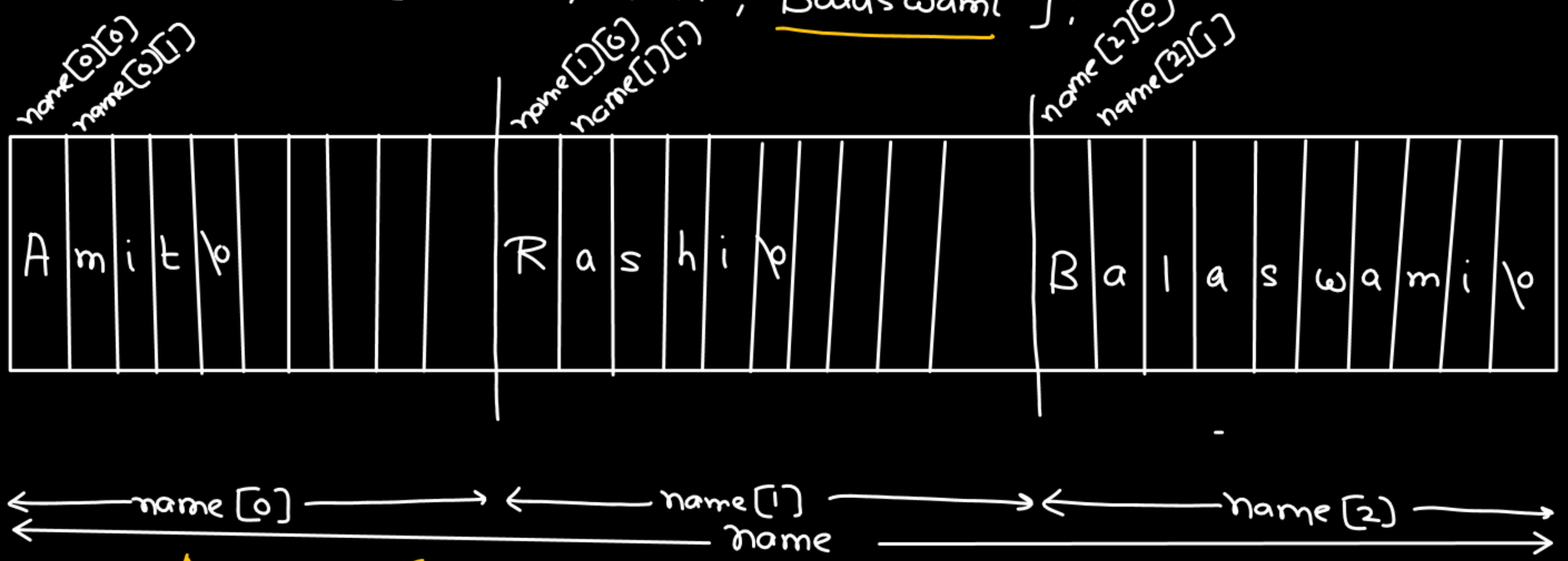
```
char arr2[10] = "Neeraj";
```
1-D array

```
char arr3[10] = "Poonam";
```
1-D array

```
char arr[3][10] = {"Pankaj", "Neeraj", "Poonam"};
```

```
char name[3][10] = { "Amit", "Rashi", "Balaswami" };
```

9 + 1



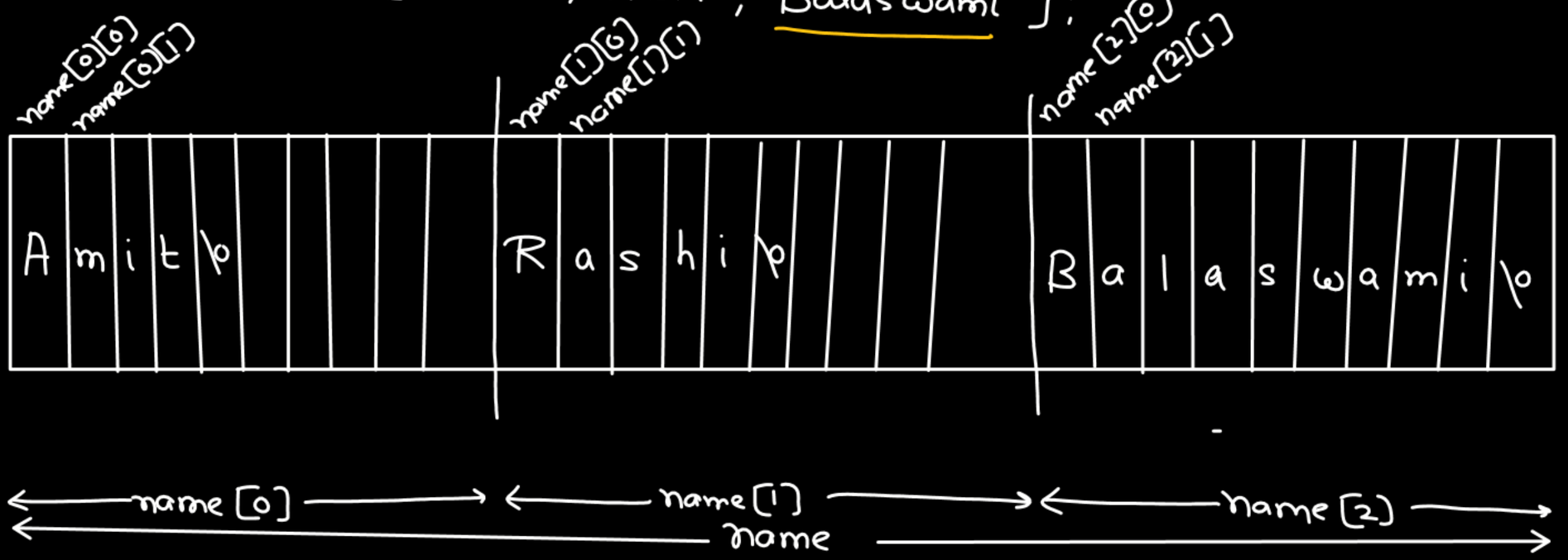
name[0] → Address ✓ ⇒ &name[0][0]

name[0] → element ✗ ⇒ Address of char 'A'

```
printf("%s", name[0]);
printf(name[0]);
```

Amit

```
Char name[3][10] = { "Amit", "Rashi", "Balaswami" };
```




```
name[1] ⇒ &name[1][0]
bf("%s", name[1]);
bf(name[1]);
bf(&name[1][0]);
```

} Rashi

name[1] ⇒ &name[1][0]

name[1]+1 ⇒ &name[1][1]

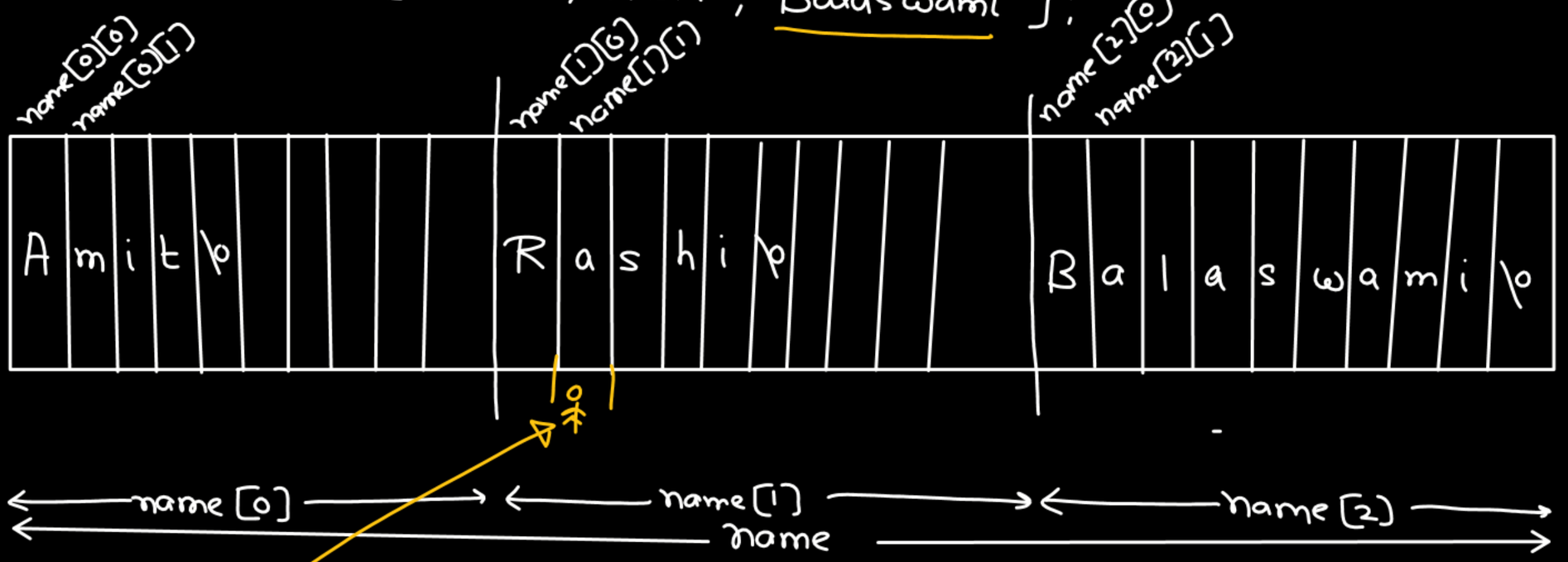
⇒ Address of 'a' in Rashi

```
bf(name[1]+1);
bf("%s", name[1]+1);
```

} ashi

Char name[3][10] = { "Amit", "Rashi", "Balaswami" };

9 + 1



name [1] + 1 ⇒ Address of 'a' in Rashi

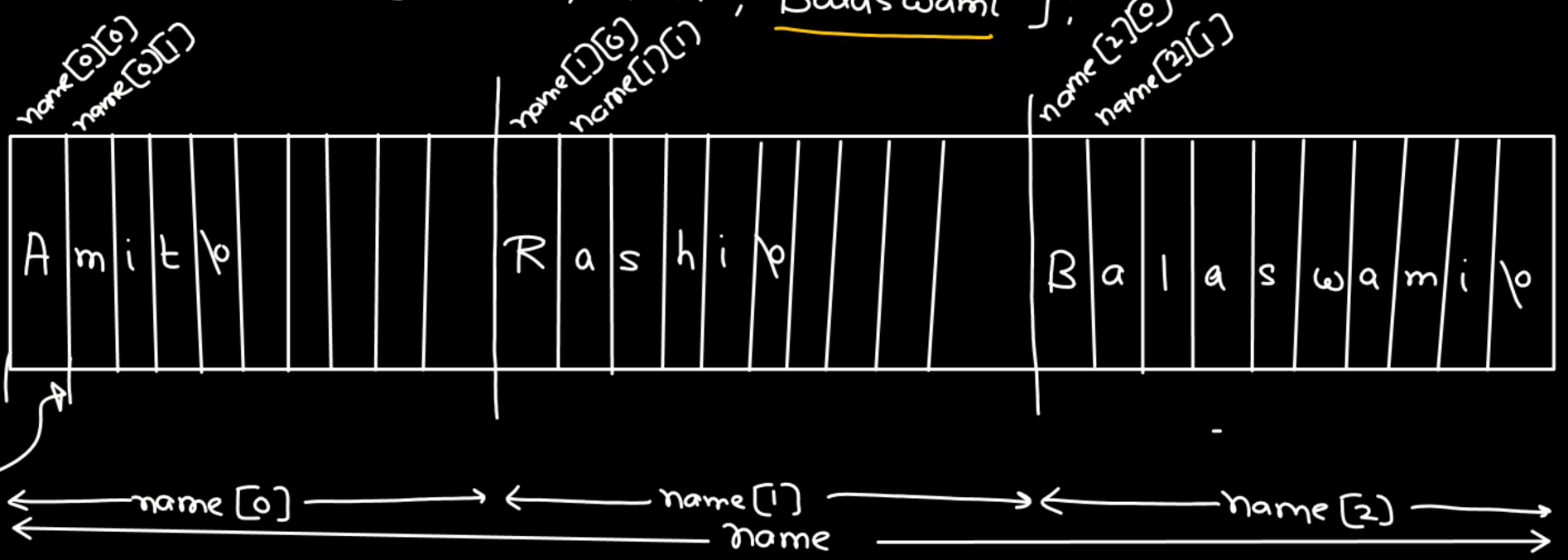
*(name [1] + 1) ⇒ 'a'

```
bf("%c", *(name[1]+1));
bf("%c", name[1][1]);
```

'a'

```
char name[3][10] = { "Amit", "Rashi", "Balaswami" };
```

9 + 1



1.) `bf( *name + 1);`

`*&name[0] + 1`

cancel

`name[0] + 1`

`&name[0][0] + 1`

mit

2.) `bf("%c", **name)`

`**name` ⇒ cancel `*&name[0]`

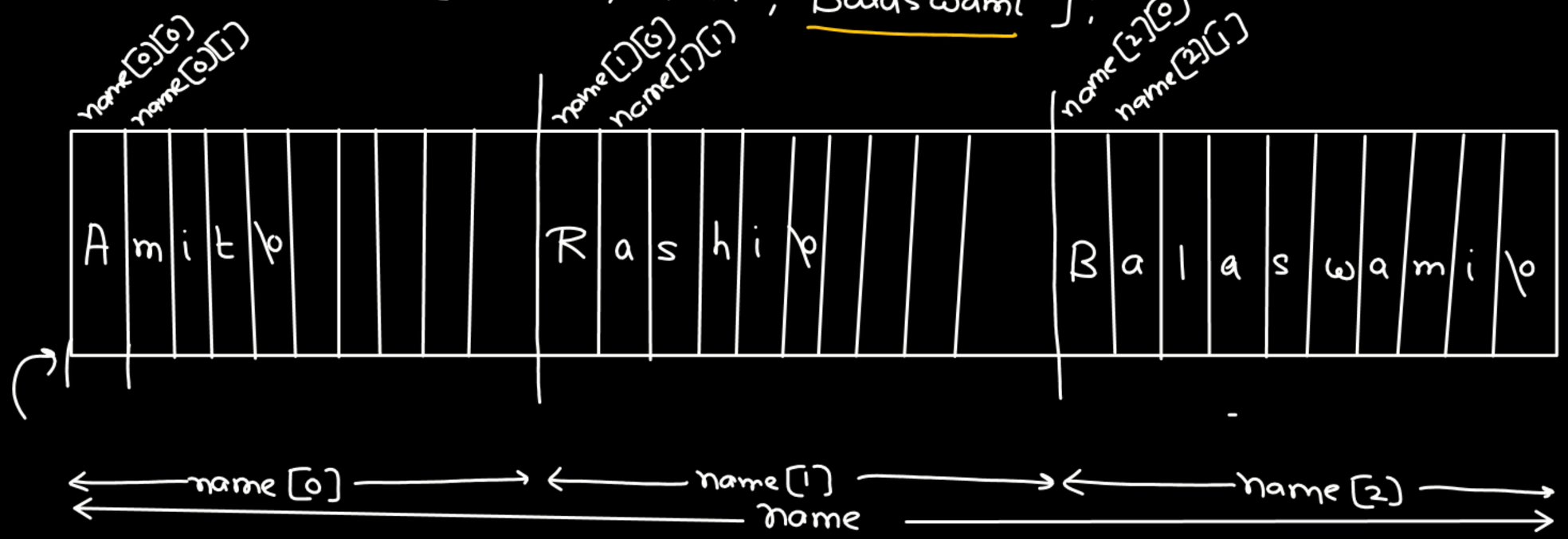
⇒ `*name[0]`

⇒ cancel `*&name[0][0]` = `name[0][0]`

A

```
char name[3][10] = { "Amit", "Rashi", "Balaswami" };
```

9 + 1




```
name[0] = "Pankaj";
```

✗ Lvalue

```
name[0][0] = 'P';
```

String

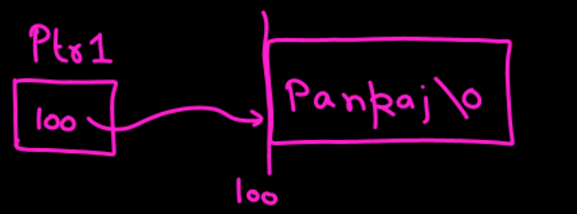
↓

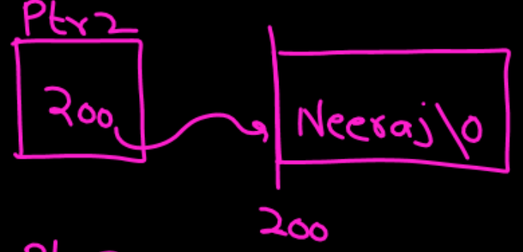
using pointer to char

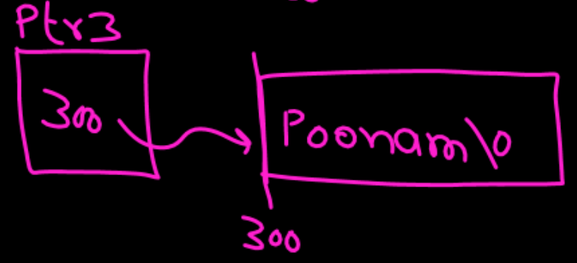
```
char *Ptr1 = "Pankaj";
```

```
char *Ptr2 = "Neeraj";
```

```
char *Ptr3 = "Poonam";
```





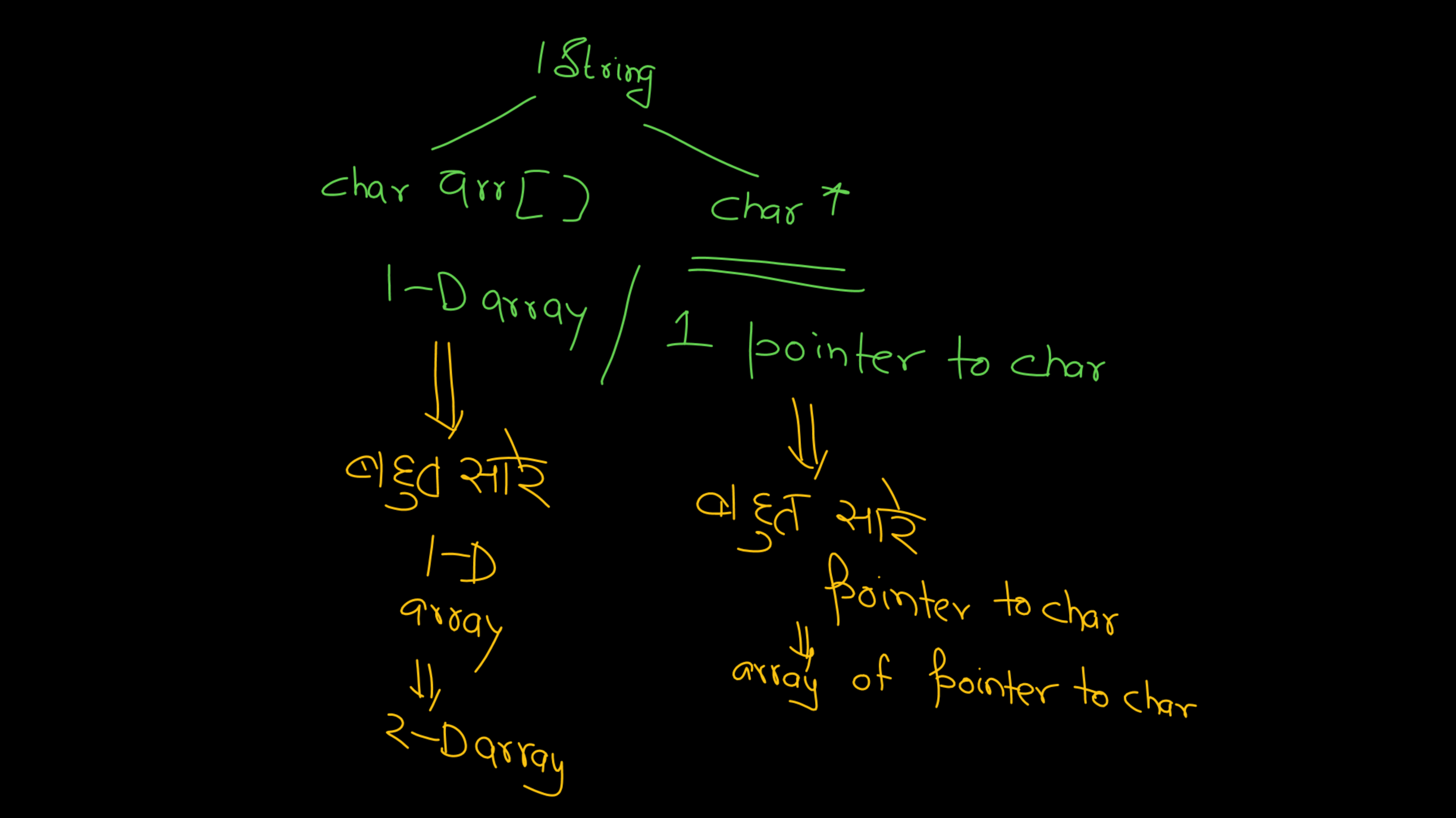


```
char *Ptr1 = "Pankaj";
char *Ptr2 = "Neeraj";
char *Ptr3 = "Amit";
```

3 pointer to char

Array of 3 pointer to char

```
char *name[3] = { "Pankaj",
                  "Neeraj",
                  "Amit" };
```


String
char arr[]
char *
1-D array
1 pointer to char
बहुत सारे
1-D
array
2-D array
बहुत सारे
pointer to char
array of pointer to char

char *name[3] = { "Amit", "Rahul", "Priya" };

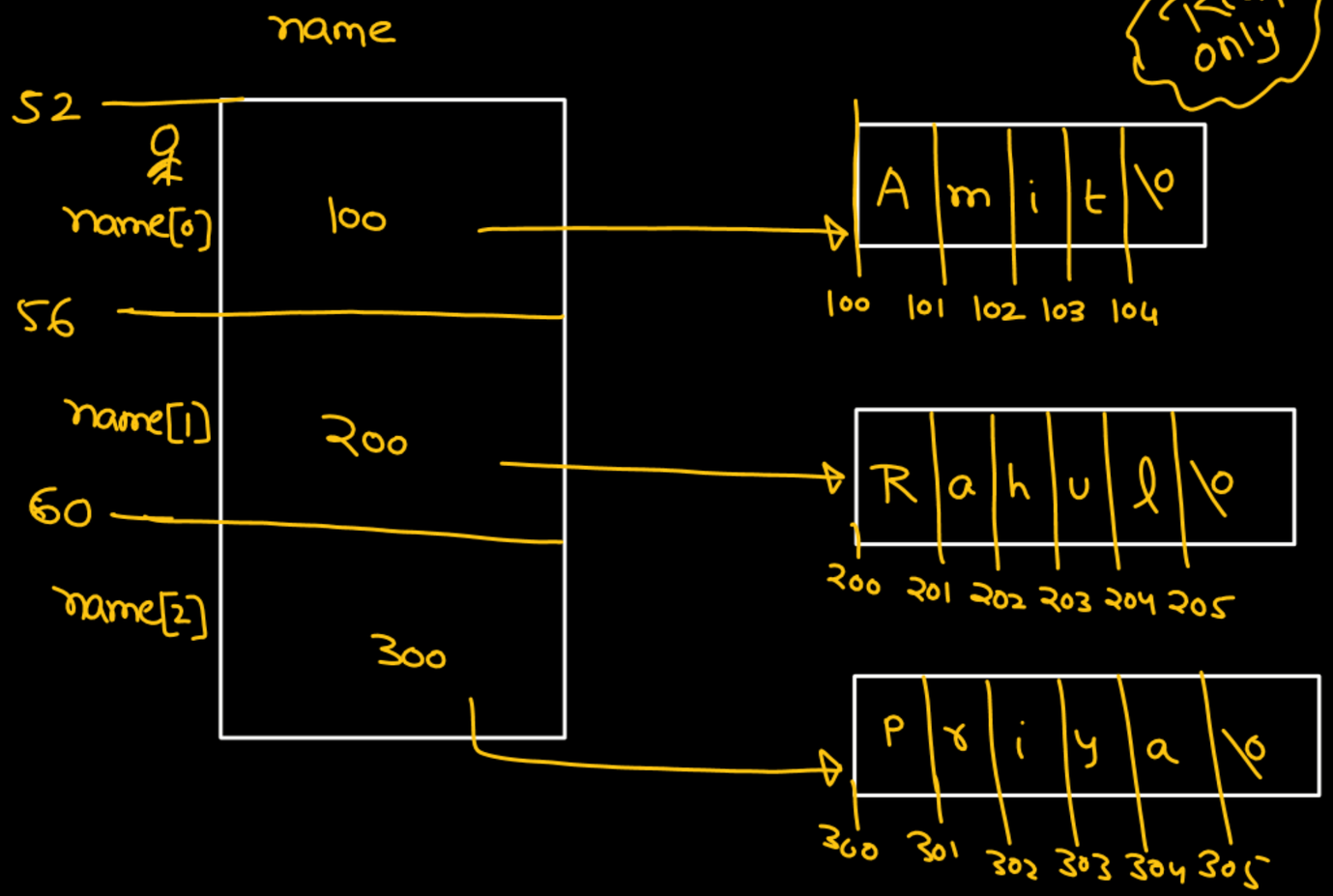
① name : &name[0] : Mem. loc. 52

② *name : *&name[0]

Read only

⇒ name[0] ⇒ Add. 100

⇒ Add of 'A' in Amit




```
printf("%s", *name);
printf("%s", name[0]);
printf(*name);
printf(name[0]);
```

} Amit

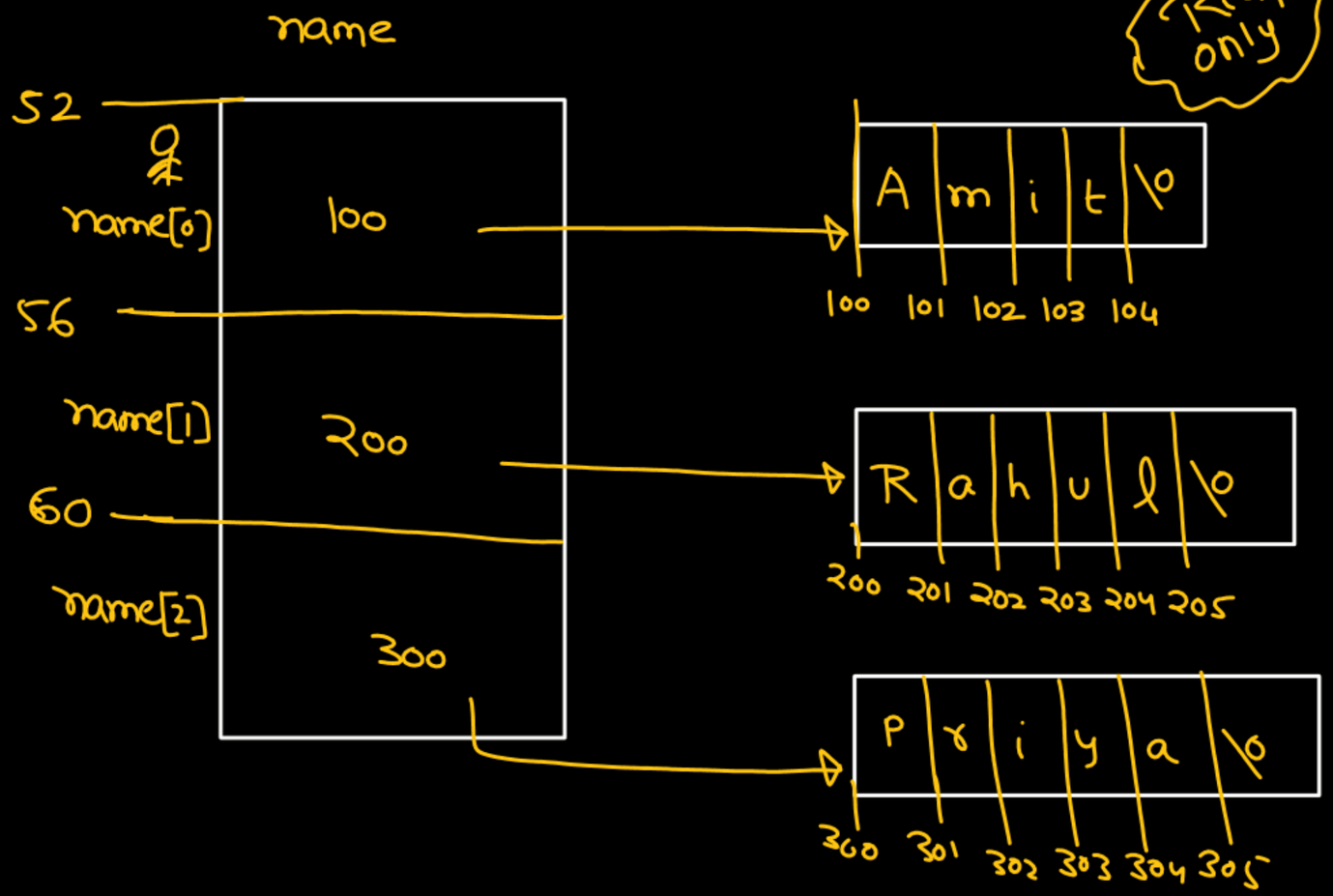
char *name[3] = { "Amit", "Rahul", "Priya" };

① name : &name[0] : Mem. loc.

② *name : *&name[0]

⇒ name[0]

Read only

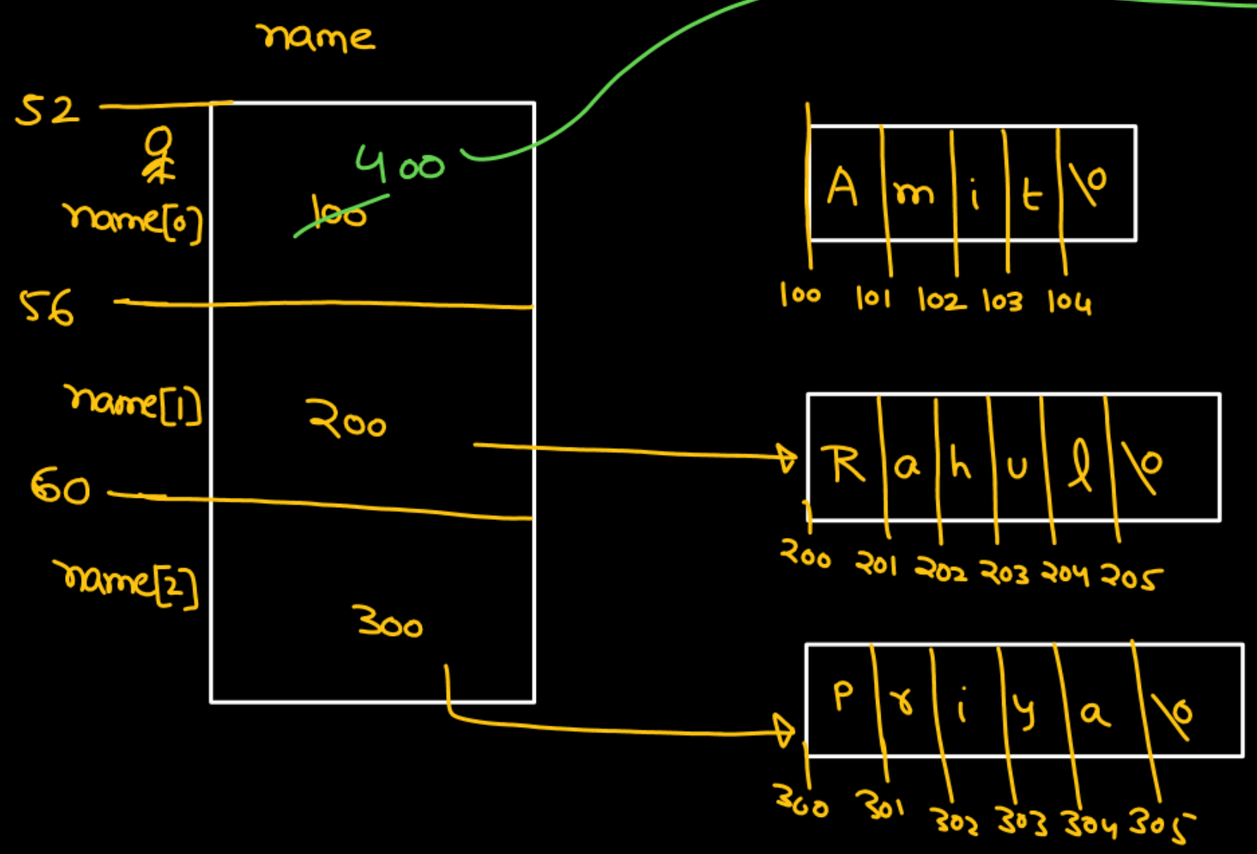


name[0]+1 ⇒ Add of 'A' +1 in Amit

⇒ Add. of 'm' in Amit

bf(name[0]+1) ⇒ mit

char *name[3] = { "Amit", "Rahul", "Priya" };

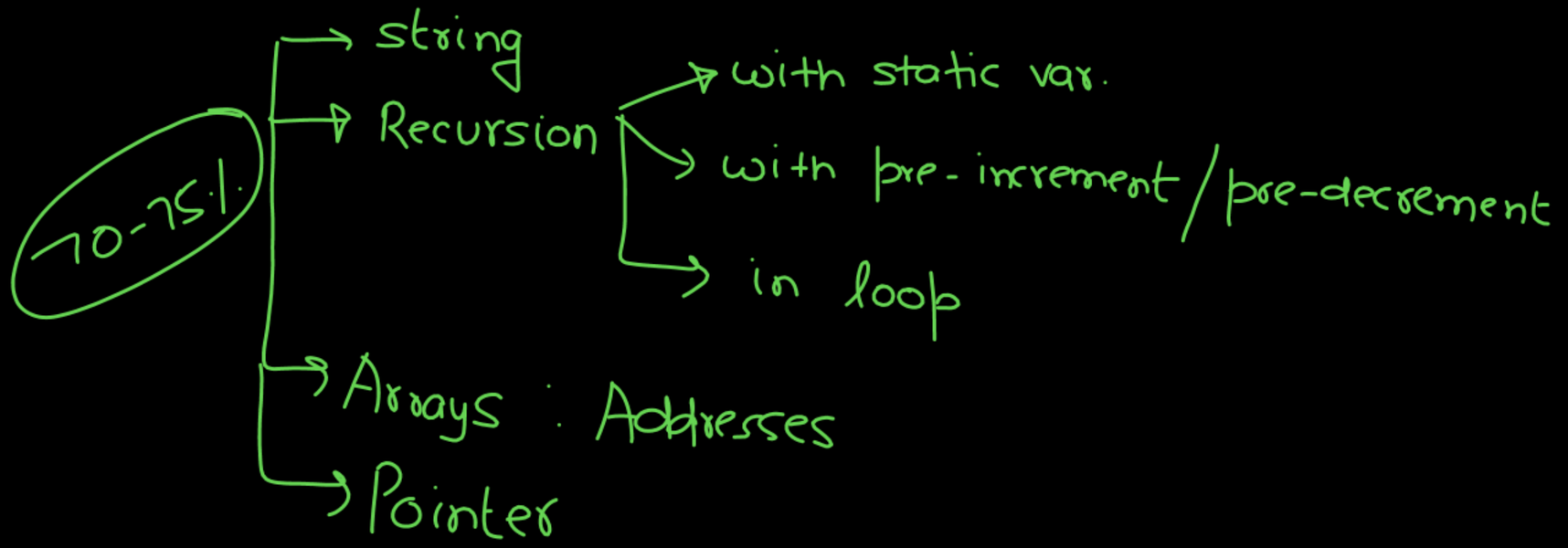


add, array, pointer, string

Revise 70% gate

name[0] = "Pankaj";

Pointer ≠


70-75:
string
Recursion
with static var.
with pre-increment / pre-decrement
in loop
Arrays : Addresses
Pointer

In-built functions : string

string.h

1) strlen( ) : To find/returns the no. of symbols/char. in the string except \0

- No. of symbol but does not count '\0'
- Count no. of symbol from beginning of the string till '\0' char. is encountered (but it does not count '\0')

```
char arr[] = "Pankaj";
```

arr → address

unsigned int strlen( )

```
char *ptr = "Pankaj";
```

Ptr → Address

```
char arr[] = "Pankaj";
int l;

l = strlen(arr);
```

char. Address

```
unsigned int strlen( [      ] char * str )
{
```

By Mistake ⇒ we make some changes in Original string

```
}
```

```
char arr[] = "Pankaj";
int l;

l = strlen(arr);
```

→ char. Address

```
unsigned int strlen( const char *str )
{

}
```

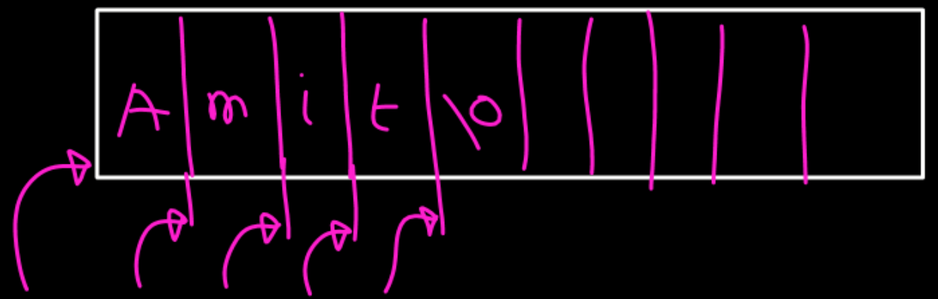
const → Can not modify actual string

By Mistake ⇒ we make some changes in Original string

unsigned int strlen(const char *)

1.

```
char arr[10] = "Amit";
int l;
l = strlen(arr);
printf("%d", l);
```

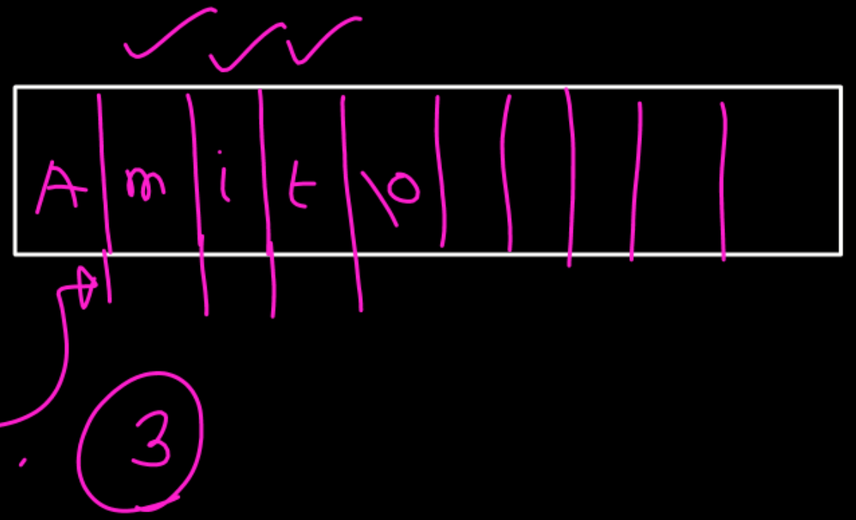


1 2 3 (4) stop

unsigned int strlen(const char *)

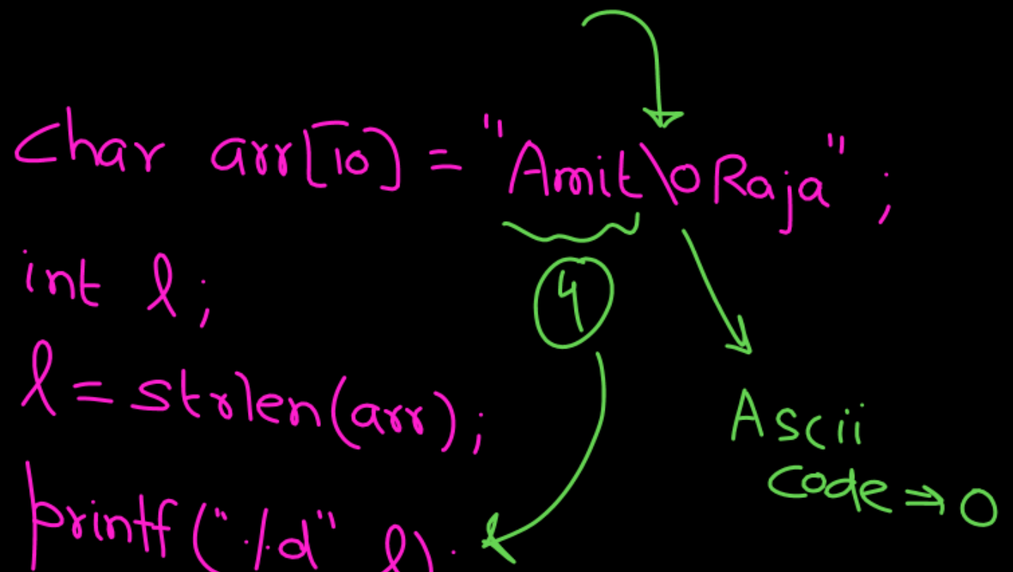
1.

```
char arr[10] = "Amit";
int l;
l = strlen(arr+1);
printf("%d", l);
```



unsigned int strlen(const char *)

1.



1.

```
char arr[10];    //Allocate memory
```

≡

but not initialize it.

```
arr = "Pankaj";
```

Invalid

strcpy → To copy a string

char* strcpy (char *destination,

const char* source)

It copy the string pointed by source pointer to the buffer pointed by destination pointer.

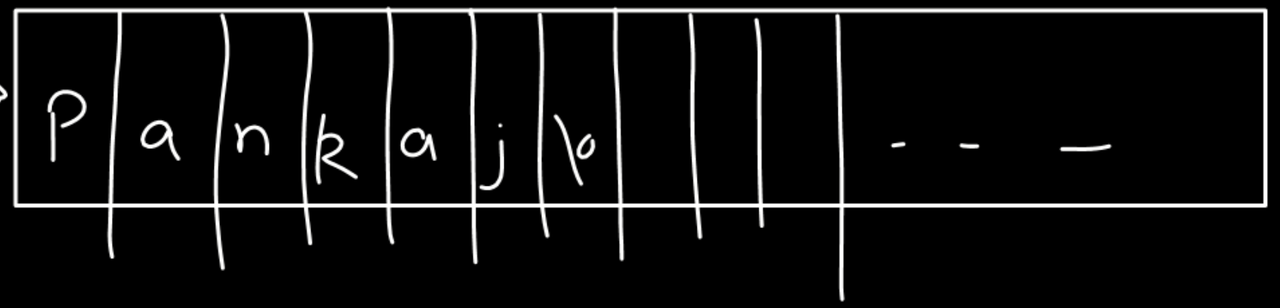
It also copies \0 char.

To avoid overflow error

→ sizeof buffer must be enough to hold string.

```
char arr[20];
```

arr




```
arr = "Pankaj"; X
```

Sol.

```
strcpy(arr, "Pankaj");
```

Biggest mistake

```
char *p;
strcpy(p, "Pankaj");
```

Error X

strcat : string concatenation

```
char* strcat(char *destination, const char *source)
```

```
char arr[20];
strcpy(arr, "Pankaj");
strcat(arr, "Sharma");
```

String pointer by destination pointer

Concate the string pointed by source pointer at the end of

H.W { working of strcmp( ) }

strlen
sizeof

Google

```
char *p;
strcat((P)"Pankaj");
```

Invalid

```
strcat {
strcpy(
```

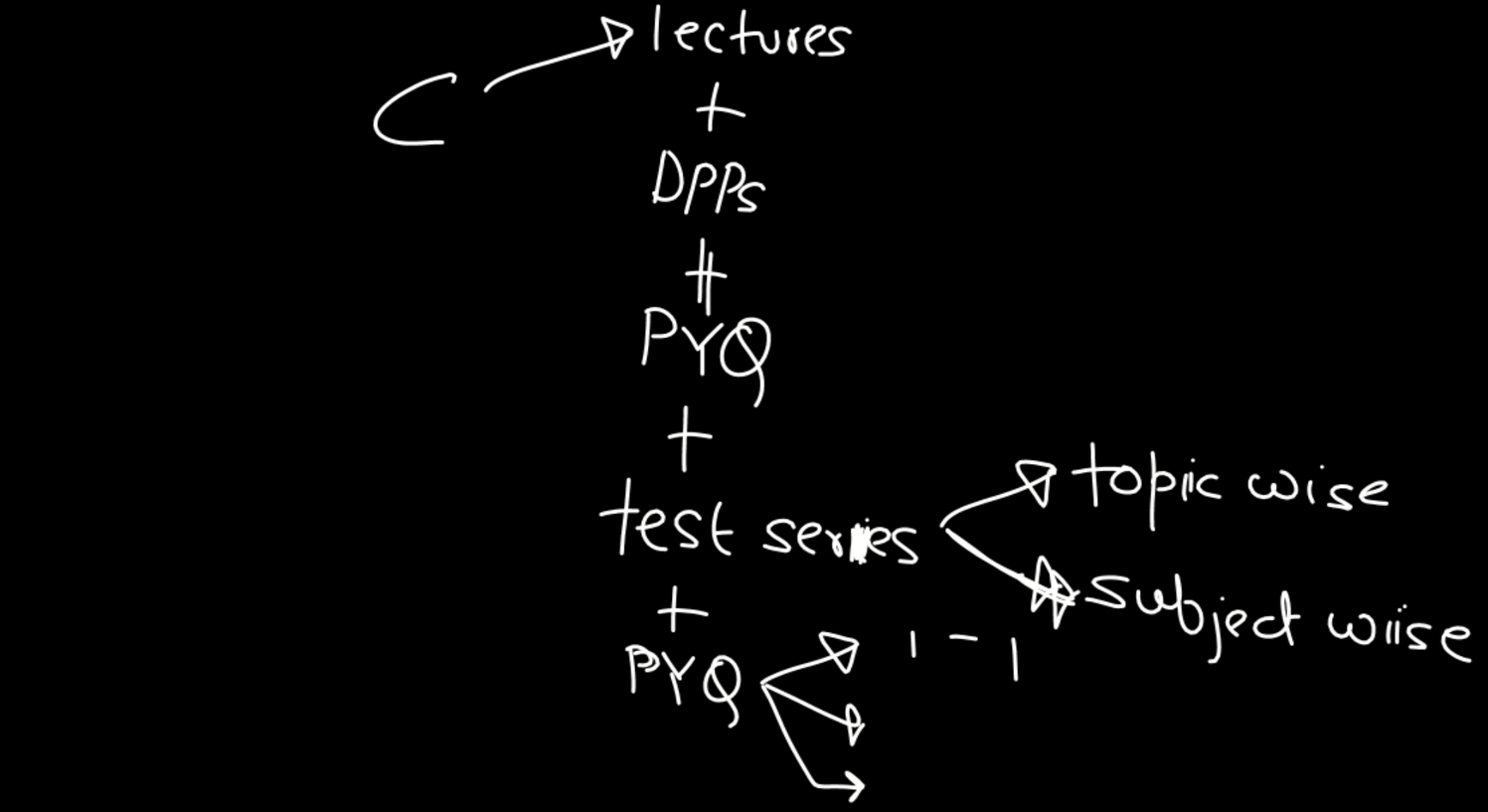
Wed : 8:30 PM

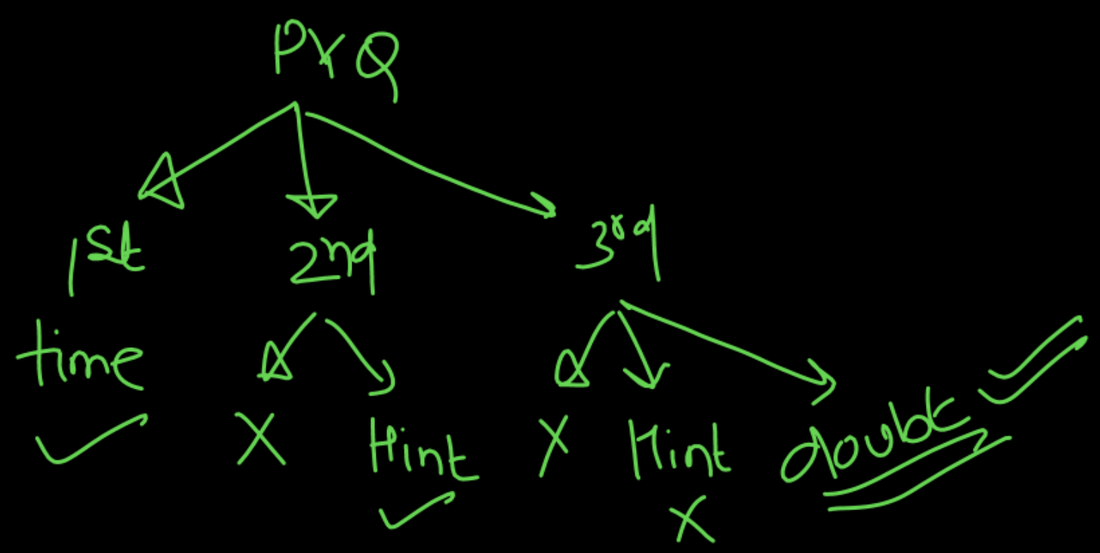
2-3 lecture

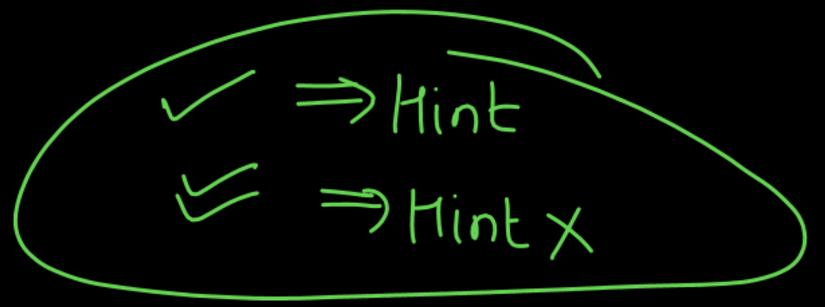
- Structure/union – 1 lecture
- Misc.
  - comma
  - sizeof
  - scoping

Data structure

- Array
- pointer
- structure


lectures
+
DPPs
+
PYQ
+
test series
topic wise
subject wise
+
PYQ
1 - 1





PYQs


THANK YOU GW
SOLDIERS !