

RED & WHITE[®]

Multimedia Education

Shaping "skills" for "scaling" higher...!!!

WELCOME, PROGRAMMERS



01.

What is sizeof() operator?

WHAT IS sizeof() operator?



sizeof() operator



The **sizeof** operator in C is used to **determine the size**, in **bytes**,
of a data type or **an object**.

It **returns** the **number of bytes** required to store an object of the
specified type.



Let's see the **syntax** of sizeof() operator...

Syntax of a sizeof() operator

`sizeof(type);`

or

`sizeof type;`



Example of sizeof() operator

```
int a = 5;  
  
printf("Size in bytes: %zu", sizeof(a));
```

```
// Output:  
Size in bytes: 4
```



%zu format specifier

The **%zu** format specifier in C is used to **print the value** of a **size_t type variable**.

The **size_t** type is an **unsigned integer type** that is used to represent the **size of objects in memory**.

We can use “**%d**” also to print size_t variables, it will not show any error.



02.

WHAT IS POINTER?

What is Pointer?



C LANG.



POINTER



A Pointer is **a variable** which **holds a memory address of another variable.**



Syntax of a Pointer

```
datatype *pointer_name;
```



Example of a Pointer

```
int *ptr;  
int a = 5;  
  
ptr = &a;  
  
printf("Address is: %u", ptr);
```

```
// Output:  
Address is: 1829548364
```



Example of a Pointer

```
int *ptr;  
int a = 5;  
  
ptr = &a;  
  
printf("Address is: %u", ptr);
```

RAM			
	a = 5		
Memory Address	1829548364	1829548368	1829548372

```
// Output:  
Address is: 1829548364
```



03.

What is Scale of Pointer?

WHAT IS SCALE OF POINTER?



SCALE OF POINTER




Scale of Pointer can be decremented or incremented as per requirement.



Example of Scale of Pointer (incremented)

```
int *ptr;  
int a = 5;  
  
ptr = &a;  
  
printf("Address is: %u", ptr+1);
```

RAM			
	a = 5		
Memory Address	1829548364	1829548368	1829548372



```
// Output:  
Address is: 1829548368
```

04.

What is Pointer with Array & String?

WHAT IS POINTER WITH ARRAY & STRING?



POINTER WITH ARRAY & STRING



A single pointer is created to hold a **base address** of an array.

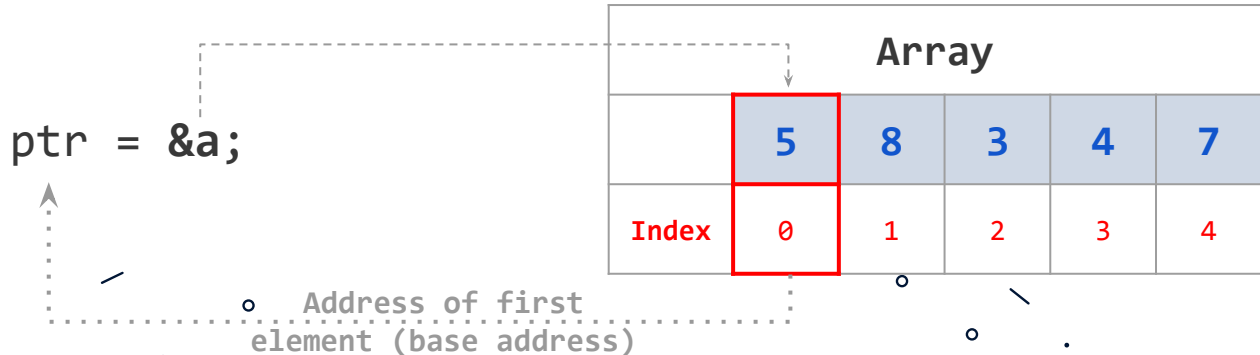
And with that **single pointer**, by **using scale of pointer**, we can **access all elements** of an array.



Use of Single Pointer for an Array

```
int *ptr;  
int a[] = {5, 8, 3, 4, 7};  
  
ptr = &a; // same as => ptr = &a[0];  
  
printf("%u => %d", ptr, *ptr);
```

```
// Output:  
1829548368 => 5
```



Using Scale of Pointer for access all elements



```
int *ptr, i;  
int a[] = {5, 8, 3, 4, 7};  
  
ptr = &a; // same as => ptr = &a[0];  
  
for(i=0; i<=4; i++)  
{  
    printf("%u => %d", ptr+i, *(ptr+i));  
}
```

```
// Output:  
1829548368 => 5  
1829548372 => 8  
1829548376 => 3  
1829548380 => 4  
1829548384 => 7
```





LANGUAGE

Let's start now...

