Lesson 23 Global Variable

Any variable that is only declared without assigning a value will automatically pick a **garbage value.**

Example:

```
#include <stdio.h>
int main() {
        int x[10], i;
        for (i = 0; i < 10; i++)
                printf("%d ", x[i]);
}
```

Try to code yourself:

- - > click here: lesson 24 Global Variable c1 - Replit

output :

-858993460 -858993460 -858993460 -858993460 -858993460
-858993460 -858993460 -858993460 -858993460 -858993460

runtime unused memory.

How to initialize an array with zero values?

- We use a **global variable** that is automatically initialized to zero.

How do we use a global variable?

- Global variables are defined before the main() function.

```
#include <stdio.h>
int x[10], i;
int main() {
        for (i = 0; i < 10; i++)
                printf("%d ", x[i]);
}
```

Try to code yourself:

- - > click here: lesson 24 Global Variable c2 - Replit

output :

0 0 0 0 0 0 0 0 0 0

All the values are 0.

Advantages of using a global variable:

- Its value is automatically initialized to **zero**.

- It can be **accessed** by **any function** in the program. On the other hand, **local variables** that are declared within a function are only accessible to that function.

Example:

```
#include <stdio.h>
void fun() {
        printf("%d", x);
        //x is only accessible to the main() function
}
int main() {
        int x = 2;
        printf("%d ", x);
        fun();
}
```

Instead, if we declared x before the main, it will be accessed by any function:

```
#include <stdio.h>
int x = 2;
```

```
void fun() {
        printf("%d", x);
}
int main() {
        printf("%d ", x);
        fun();
}
```

output:

2 2

There are no warnings and no errors because x is a **global variable.**