The name **malloc** and calloc() are library functions that allocate memory dynamically. It means that memory is allocated during runtime(execution of the program) from heap segment.

Initialization:
```
void * malloc( size_t size );
```
malloc() allocates memory block of given size (in bytes) and returns a pointer to the beginning of the block. malloc() doesn't initialize the allocated memory. If we try to acess the content of memory block then we'll get garbage values.
calloc() allocates the memory and also initializes the allocates memory block to zero. If we try to access the content of these blocks then we'll get 0.

Number of arguments: Unlike malloc(), calloc() takes two arguments:
```
void * calloc( size_t num, size_t size );
```
Number of blocks to be allocated.
Size of each block.

Return Value: After successful allocation in malloc() and calloc(), a pointer to the block of memory is returned otherwise **NULL** value is returned which indicates the failure of allocation.

```c
// C program to demonstrate the use of calloc()
// and malloc()
#include <stdio.h>
#include <stdlib.h>

int main()
{
   int *arr;

   // malloc() allocate the memory for 5 integers
   // containing garbage values
   arr = (int *)malloc(5 * sizeof(int)); // 5*4bytes = 20 bytes

   // Deallocates memory previously allocated by malloc() function
   free( arr );

   // calloc() allocate the memory for 5 integers and
   // set 0 to all of them
   arr = (int *)calloc(5, sizeof(int));

   // Deallocates memory previously allocated by calloc() function
   free(arr);

   return(0);
}
```

We can achieve same functionality as calloc() by using malloc() followed by memset().

```
ptr = malloc(size);
memset(ptr, 0, size);
```

Note:

It would be better to use malloc over calloc, unless we want the zero-initialization because malloc is faster than calloc. So if we just want to copy some stuff or do something that doesn't require filling of the blocks with zeros, then malloc would be a better choice.

Thank you for watching!

Please leave us your comments.