What is malloc in C?

The malloc() function stands for memory allocation. It is a function which is used to allocate a block of memory dynamically. It reserves memory space of specified size and returns the null pointer pointing to the memory location. The pointer returned is usually of type void. It means that we can assign malloc function to any pointer.

Syntax

```
ptr = (cast_type *) malloc (byte_size);
```

Here,

- ptr is a pointer of cast_type.
- The malloc function returns a pointer to the allocated memory of byte_size.

```
Example: ptr = (int *) malloc (50)
```

When this statement is successfully executed, a memory space of 50 bytes is reserved. The address of the first byte of reserved space is assigned to the pointer ptr of type int.

Consider another example of malloc implementation:

```
#include <stdlib.h>
int main(){
int *ptr;
ptr = malloc(15 * sizeof(*ptr)); /* a block of 15 integers */
    if (ptr != NULL) {
      *(ptr + 5) = 480; /* assign 480 to sixth integer */
      printf("Value of the 6th integer is %d",*(ptr + 5));
    }
}
```

Output:

```
Value of the 6th integer is 480
```

What is calloc in C?

The **calloc()** in C is a function used to allocate multiple blocks of memory having the same size. It is a dynamic memory allocation function that allocates the memory space to complex data structures such as arrays and structures and returns a void pointer to the memory. Calloc stands for contiguous allocation.

Malloc function is used to allocate a single block of memory space while the calloc function in C is used to allocate multiple blocks of memory space. Each block allocated by the calloc in C programming is of the same size.

calloc() Syntax:

```
ptr = (cast_type *) calloc (n, size);
```

- The above statement example of calloc in C is used to allocate n memory blocks of the same size.
- After the memory space is allocated, then all the bytes are initialized to zero.
- The pointer which is currently at the first byte of the allocated memory space is returned.

Whenever there is an error allocating memory space such as the shortage of memory, then a null pointer is returned as shown in the below calloc example.

How to use calloc

The below calloc program in C calculates the sum of an arithmetic sequence.

```
#include <stdio.h>
    int main() {
        int i, * ptr, sum = 0;
        ptr = calloc(10, sizeof(int));
        if (ptr == NULL) {
            printf("Error! memory not allocated.");
            exit(0);
        }
        printf("Building and calculating the sequence sum of the first 10 terms
\ n ");
        for (i = 0; i < 10; ++i) { * (ptr + i) = i;
            sum += * (ptr + i);
        }
        printf("Sum = %d", sum);
        free(ptr);
        return 0;
    }
```

Result of the calloc in C example:

```

Building and calculating the sequence sum of the first 10 terms
Sum = 45
```

Syntax

The syntax for the free function in the C Language is:

```
void free(void *ptr);
```

Parameters or Arguments

ptr

The pointer to the memory block to release.

Note

- The block of memory must have been allocated by one of the following functions - calloc function, malloc function, or realloc function.

Returns

The free function does not return anything.

Syntax

The syntax for the realloc function in the C Language is:

```
void *realloc(void *ptr, size_t size);
```

Parameters or Arguments

ptr

The old block of memory.

size

The size of the elements in bytes.

Note

- *ptr* must have been allocated by one of the following functions - calloc function, malloc function, or realloc function.

Returns

The realloc function returns a pointer to the beginning of the block of memory. If the block of memory can not be allocated, the realloc function will return a null pointer.

Required Header

In the C Language, the required header for the realloc function is:

```
#include <stdlib.h>
```