Dynamic Memory Allocation

Dynamic Memory Allocation

- It is a procedure in which the size of a data structure is changed during the runtime.
- There are 4 library functions provided by C, defined under **<stdlib.h>** header file to facilitate DMA.
- These four functions are-

✔ malloc()
✔ calloc()
✔ realloc()
✔ free()

MALLOC()

- **“malloc”** or **“memory allocation”** method in C is used to dynamically allocate a single large block of memory with the specified size.
- It returns a pointer of type void, which can be cast into a pointer of any form.
- It initializes each block with default garbage value.

MALLOC()

- **Syntax:**

ptr = (cast-type*) malloc(byte-size)

- **Example:**

ptr = (int*) malloc(sizeof(int));

Since the size of int is 4 bytes, this statement will allocate 4 bytes of memory. And, the pointer ptr holds the address of the first byte in the allocated memory.

CALLOC()

- **“calloc”** or **“contiguous allocation”** method in C is used to dynamically allocate the specified number of blocks of memory of the specified type.
- It initializes each block with a default value ‘0’.
- It returns NULL if memory is not sufficient.

CALLOC()

- **Syntax:**

ptr = (cast-type*)calloc(n, element-size);

- **Example:**

ptr = (float*) calloc(25, sizeof(float));

This statement allocates contiguous space in memory for 25 elements each with the size of the float.

MALLOC() VS CALLOC()

malloc()	calloc()
malloc() will create a single block of memory of size specified by the user.	calloc() can assign multiple blocks of memory for a variable.
malloc function contains garbage value.	The memory block allocated by a calloc function is always initialized to zero.
malloc() is faster than calloc().	calloc() is slower than malloc().
Time efficiency is higher than calloc().	Time efficiency is lower than malloc().
malloc() returns only starting address and does not make it zero.	Before allocating the address, calloc() returns the starting address and make it zero.
It does not perform initialization of memory.	It performs memory initialization.

REALLOC()

- **“realloc”** or **“re-allocation”** method in C is used to dynamically change the memory allocation of a previously allocated memory.

- If the memory previously allocated with the help of malloc() or calloc() is insufficient, realloc() can be used to **dynamically re-allocate memory**.

- Re-allocation of memory maintains the already present value and new blocks will be initialized with default garbage value.

REALLOC()

- **Syntax:**

ptr = realloc(ptr, newSize);

- **Example:**

ptr = realloc(ptr, n * sizeof(int));

This statement will add another n number of memory locations with the previously allocated ones.

FREE()

- **“free”** method in C is used to dynamically **de-allocate** the memory.
- The memory allocated using functions malloc() and calloc() is not de-allocated on their own.
- Hence the free() method is used, to reduce wastage of memory by freeing it.

FREE()

- **Syntax:**

free(ptr);

- **Example:**

free(ptr2);