DYNAMIC MEMORY ALLOCATION

DYNAMIC MEMORY ALLOCATION

- Size of array is fixed so memory may be insufficient or may be wasted.
- Consider the following scenarios :
  - Dimension of 1D array is not known.
  - For 2D Array column size is not known or both the dimensions are not known.
- Dynamic Memory Allocation allows memory allocation at runtime.
- Built-in Functions stored in **stdlib.h** are **:**

➢**malloc()**

➢**calloc()**

➢**realloc()**

malloc() Memory Allocation

- The following is prototype of malloc() function

  void* malloc(size_t num_bytes);

- To **malloc()** we need to pass the number of bytes to allocate in memory. On doing so, it allocates **single large block of memory** and return void pointer if successful, which can be casted to whatever type required.
- If **malloc()** is unable to allocate requested amount of memory then it returns NULL pointer.
- Syntax:

  int ptr = (cast-type*)malloc(size_in_bytes);

calloc() Continuous Memory Allocation

- It is very much similar to **malloc().**
- To **calloc()** we need to pass total no of elements and size of each element. On doing so, it allocates **specified number of blocks of memory** and return void pointer if successful.
- If **calloc()** is unable to allocate requested amount of memory then it returns NULL pointer.
- Syntax:

int *ptr = (cast-type*)calloc(total_elements, element_size);

realloc() Re-allocation of Memory

- It is used to dynamically change the memory allocation of previously allocated memory. It appends the new memory to existing memory block.
- If memory previously allocated with the help of **malloc()** and **calloc()** is insufficient, **realloc()** can be used to dynamically reallocate the memory.
- Syntax:

  int *ptr = realloc (ptr, size_in_bytes);
- The function changes the size of space pointed by **ptr**. The new memory could be adjacent to existing block or entirely new memory location is allocated depending on availability.

free()

- **free()** frees the specified memory block to be used for another purpose.
- It dynamically de-allocates the memory allocated using **malloc()** and **calloc()**.
- Syntax:

  free(ptr);
- It is responsibility of programmer to free dynamically allocated memory. If not done, then there will be memory leakage.