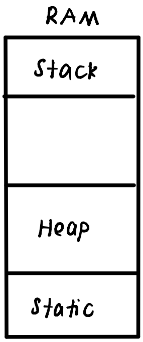
Memory Management



Volatile memory is divided into 3 sections:

1. static
2. stack
3. heap

Static memory: Used for storing global variables and variables designated as 'static' in code.

Stack memory: used for automatic allocation of local variables

: organized as a last-in-first-out (LIFO) system so the variables of one function can be "pushed" to the stack when new function is called.

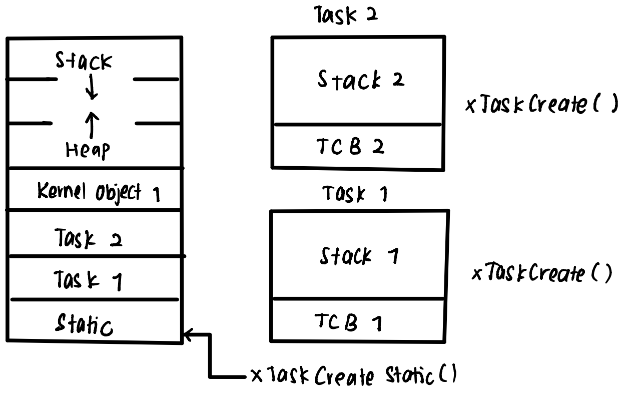
: while returning to the first function, that functions' variables can be "popped" off, which the function can use to continue running where it left off.

Heap memory : Allocated explicitly known as dynamic allocation

: malloc () function is used to allocate heap of the variables, buffers, etc.

: In C or C++ , heap memory must be deallocate when it is no longer used. Otherwise, it will result in a memory leak and causing some effects (corrupting other part of memory).

- Stack and heap grow toword each other and taking up unallocated memory where needed. They collide and begin overwriting each other data when it is left unchecked.
- Creating a task in RTOS, the operating system will allocate a section of heap memory for the tosk.



- Task Control Block (TCB) used to store information about the task (priority and local stack pointer).
  (one part of allocated memory)
- The other part reserved as a local stack that operates just like a global stack
- Local variables created during function calls within a task are pushed to the task's local stack.
- It is important to calculate the predicted stack usage of a task ahead of time and include it as the stack size parameter in xTaskCreate ().