Data segment is a portion of virtual address space of a program containing the global variables and static variables that are initialized by the programmer. The values of variables can be changed at running time.

Function ***malloc()*** is a standard C Library function which is used to allocate the dynamic memory. Dynamic allocation means you allocate the memory at run time, when you do not know the amount of memory during compile time. The C standard guarantees that ***malloc()*** will return a suitably aligned memory block for any of the standard types but there may be situations where you want stricter alignment. For example, the address of a block returned by ***malloc()*** in GNU systems is a multiple of 8 so if we need a block whose address is a multiple of higher power of 2, then use ***aligned_malloc***.

The function ***malloc()*** is to allocate memory dynamically while ***free()*** is to deallocate and return the memory to the heap. Those function will manage the size of heap by expanding and shrinking it as needed, so the question is *"Can we increase size of heap in running process?"*. As we knew, the end of heap is marked by a *"break"* pointer. When you want to increase the size of heap, we have to move the "break" further. There are system calls *"brk"* and *"sbrk"* can do those works. *"brk"* and *"sbrk"* define the end of the process's data segment. *"brk()"* sets the end of the data segment by address when the system has enough memory while *"sbrk()"* increases the data space by increment bytes.