1. BIOS:
   1. Stands for **basic input/output system**
   2. BIOS comes included with computers, as firmware on a chip on the motherboard.
   3. Tasks carried out by BIOS:
      1. **Power-on self-test (POST):** This tests the hardware of the computer before loading the OS.
      2. **Bootstrap loader:** This locates the OS.
      3. Software/drivers: This locates the software and drivers that interface with the OS once running.
   4. Its two major procedures or functions are determining what peripheral devices (keyboard, mouse, disk drives, printers, video cards, etc.) are available and loading the operating system (OS) into main memory.
   5. BOOT SEQUENCE:
      1. CPU runs an instruction in memory for the BIOS.
      2. Power-on self-test
      3. BIOS finds a bootable device.
      4. BIOS reads the first sector (boot sector) from bootable device into memory.

The boot sector is known as first sector because it has a specific address: Cylinder 0, Head 0, Sector 1.

The information in the first sector is known as Master Boot Record (MBR).

* + - 1. MBR contains the information regarding how and where the Operating system is located in the hard disk so that it can be booted in the RAM.
      2. The size of MBR is commonly less than or equal to 512 bytes.
      3. MBR is sometimes called **master partition table** because it includes a partition table which locates every partition in the hard disk. MBR contains programs that determine which partition on the hard disk is used for the system boot.
    1. BIOS loads MBR from bootable device into 0x7C00 memory address.
  1. BIOS can now be replaced by UEFI
     1. UEFI stands for Unified Extensible Firmware Interface.
     2. It does the same job as a BIOS, but with one basic difference: it stores all data about initialization and start-up in an .efi file, instead of storing it on the firmware.
     3. This .efi file is stored on a special partition called EFI System Partition (ESP) on the hard disk. This ESP partition also contains the bootloader.
     4. Difference between BIOS (Legacy) and UEFI:
        1. UEFI supports bootable drive sizes up to 9 zettabytes, whereas BIOS only supports 2.2 terabytes.
        2. UEFI provides faster boot time.
        3. UEFI provides secure boot which means UEFI can allow only authentic drivers and services to load at boot time, making sure that no malware can be loaded at computer start-up. Microsoft implemented this feature in windows 8 and 10 to counter piracy issues in Windows, while Mac has been using UEFI for quite some time now.

NOTE: We can check whether we have BIOS Legacy mode or UEFI mode by typing ‘msinfo’ in search bar.

* + - 1. We might not need UEFI in this project because:
         1. We’re beginner and don't care about messing with any type of firmware, BIOS is for us.
         2. We have < 2 TB of bootable partition.

1. Memory Addressing and how memory address is formed in real mode:
   1. There are 6 special registers called **segment registers** which are 16-bit registers.

But in this project, we will only use 4 segment registers:

* + 1. cs **Code Segment**

🡪 It stores the base location of the code section (.text section) which is used for data access.

* + 1. ds **Data Segment**

🡪 It stores the default location for variables (.data section) which is used for data access.

* + 1. es **Extra Segment**

🡪 It is used during string operations.

* + 1. ss **Stack Segment**

🡪 It stores the base location of the stack segment and is used when implicitly using the stack pointer or when explicitly using the base pointer.

* 1. The format of an address is the

segment register: Offset 🡺 logical address

segment register X 16 + Offset 🡺 physical address

* 1. **General Purpose Registers:**
     1. 8-bit : al ah bl bh
     2. 16-bit : ax bx cx dx
     3. 32-bit : eax ebx ecx edx
     4. 64-bit : rax rbx rcx rdx