Program - > .cpp file -> compiler -> compile-> executable == Program

Program resides in disk

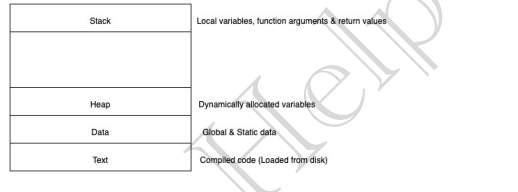
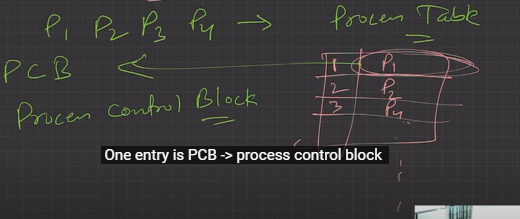
Program eg. Tiktok > OS-> program -> process

Process – Program under execution

Why Process?

* User – work ->. Way
* Program to process journey
  + How OS Creates a process

1. Load the program and static data to memory(Static data - > used for initialisation) eg. Char\* name = “Dhanesh”
2. Allocate Run time stack
   1. Stack - Part of memory used for local variables, function arguments and return values
3. Allocate Heap: to allocate dynamic allocations
   1. Part of memory used for dynamic allocations
4. IO tasks:
   * 1. UNIX
        1. I/p - handle
        2. o/p - output handle: where the data will go
        3. error – err handle fprintp(stderr,”hang”) for handling errors
5. Os handsoff control to main
   1. Return 0 – it denotes that program has been successfully executed
   2. Exit(-1) – some error has occurred

* Architecture of process
  + 
  + Text – compiled code loaded from disk
  + Data – C++ global variables or main func variables or static variable
  + Heap – Dynamic Allocated execution
  + Stack - local variables, function arguments and return values
    - Errors:
      * Stack over flow : if there is no proper base condition to program execution. The stack size gets increasing until it hits heap
        + How to avoid : Stop using proper base case intelligently
      * 2. Out of memory: whenever after memory allocation, deallocation does not take place. So error occurs when heap memory meets stack mem
        + How to avoid: Deallocate unnecessary objects
* Attributes of process:
  + Each record table of Process table is called process control block
  + Every process has its own personal pcd, os recognises process through pCB
  + 
  + PCB in itself is a data structure:
    - Process ID – Unique Identifier. It is used to identify between different processes
    - Program counter: Program is a collection of instructions
      * Program counter starts from 0. For next instruction it get incremented. It store next instruction address in PC
      * Fetch the instruction from address in PC -> PC++ -> Execute the instruction -> repeats until program terminates
* Process state : New, Wait, RUN state
* Priority: Entry to decide the priority of process to be executed
* Register: Stack pointer register, base pointer register, control registers, etc. Saves the registers during context switching and then restored when needed
* List of open files – files which are open
* Open device list – Devices open