**Stack**-> function parameters & local variables.

**heap**-> allocates memory during runtime.

**data**->static variable or global variable.

**text section**-> programs.

**What Is a Program?**

Program is an executable file containing the set of instructions written to perform a specific job on your computer.

 For example, **chrome.exe** is an executable file containing the set of instructions written so that we can view web pages. **notepad.exe** is an executable file containing the set of instructions which help us to edit and print the text files.

Programs are not stored on the primary memory in your computer. They are stored on a disk or a secondary memory on your computer. They are read into the primary memory and executed by the kernel. A program is sometimes referred as **passive entity** as it resides on a secondary memory.

**What Is a Process?**

Process is an executing instance of a program.

For example, when you double click on the Google Chrome icon on your computer, you start a process which will run the Google Chrome program. When you double click on a notepad icon on your computer, a process is started that will run the notepad program.

A process is sometimes referred as **active entity** as it resides on the primary memory and leaves the memory if the system is rebooted. Several processes may relate to same program. For example, you can run multiple instances of a notepad program. Each instance is referred as a process.

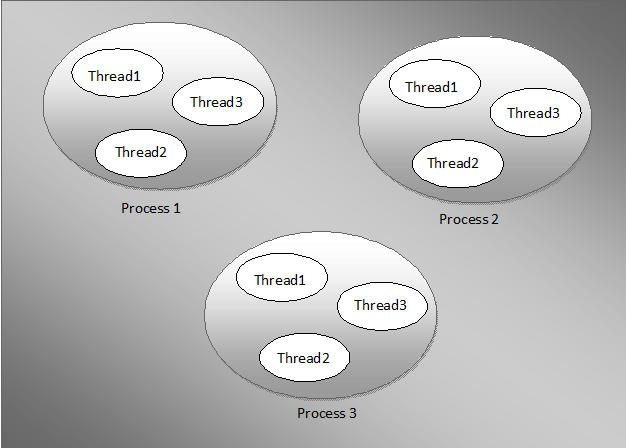
**What Is a Thread?**

Thread is the smallest executable unit of a process.

For example, when you run a notepad program, operating system creates a process and starts the execution of main thread of that process.

A process can have multiple threads. Each thread will have their own task and own path of execution in a process. For example, in a notepad program, one thread will be taking user inputs and another thread will be printing a document.

All threads of the same process share memory of that process. As threads of the same process share the same memory, communication between the threads is fast.



**Process Vs Thread :**

|  |  |
| --- | --- |
| Process | Thread |
| Processes are heavy weight operations. | Threads are light weight operations. |
| Every process has its own memory space. | Threads use the memory of the process they belong to. |
| Inter process communication is slow as processes have different memory address. | Inter thread communication is fast as threads of the same process share the same memory address of the process they belong to. |
| Context switching between the process is more expensive. | Context switching between threads of the same process is less expensive. |
| Processes don’t share the memory with other processes. | Threads share the memory with other threads of the same process. |

**Can we have multiple main methods in same class in Java?**

From the above program, we can say that **Java can have multiple main methods but with the concept of overloading**. There should be only one main method with parameter as string[ ] arg. Check MultipleMain class in intelliji.

## Types of Priority

In Java, a thread's priority is an integer in the range 1 to 10. The larger the integer, the higher the priority. The thread scheduler uses this integer from each thread to determine which one should be allowed to execute. **The Thread class defines three types of priorities**:

* Minimum priority
* Normal priority
* Maximum priority

The Thread class defines these priority types as constants MIN\_PRIORITY, NORM\_PRIORITY, and MAX\_PRIORITY, with values 1, 5, and 10, respectively. **NORM\_PRIORITY is the default priority for a new Thread**.