20TH JULY MULTI-THREADING

1. ***What do you mean by Multithreading? Why is it important?***
   Ans: *Multithreading refers to the ability of a program or process to execute multiple threads concurrently. A thread is a lightweight unit of a process that can be scheduled and executed independently. Each thread represents a separate flow of execution within the same program, allowing multiple tasks to be performed simultaneously. Multithreading is important because it can lead to significant improvements in the performance and responsiveness of applications. By using multiple threads, a program can efficiently utilize the available CPU resources and handle concurrent tasks more effectively, such as handling user interfaces while performing background tasks or processing multiple data streams simultaneously*

2. ***What are the benefits of using Multithreading?***
   Ans: *The benefits of using Multithreading include:*
   a) Improved performance: By executing tasks concurrently, a program can make better use of available CPU resources, leading to faster execution times.
   b) Enhanced responsiveness: Multithreading allows applications to remain responsive to user input while performing time-consuming tasks in the background.
   c) Resource sharing: Threads within the same process can share resources, such as memory, which can lead to efficient memory usage.
   d) Simplified programming model: In some cases, using threads can simplify the design and implementation of certain types of applications, especially those that require concurrent operations.

3. ***What is Thread in Java?***

   Ans: *In Java, a Thread is an object that represents a single thread of execution within a Java program. It is an instance of the java.lang.Thread class and is used to create and manage threads in a Java application. Threads are essential in Java for achieving concurrency and parallelism.*

4. ***What are the two ways of implementing thread in Java?***
   Ans: *There are two ways to implement threads in Java:*
   a) Extending the Thread class: This involves creating a new class that extends the Thread class and overrides the run() method to specify the code that the thread will execute. Then, an instance of this custom class is created and started using the start() method.
   b) Implementing the Runnable interface: This approach involves creating a class that implements the Runnable interface and provides the implementation for the run() method. The run() method's code is then executed by a thread when the Runnable object is passed to a Thread instance and started using the start() method.

5. ***What's the difference between thread and process?***
   Ans: *The difference between a thread and a process lies in their fundamental characteristics:*

Thread: A thread is a lightweight unit of a process. Multiple threads share the same memory space and resources of the process they belong to. Each thread has its own program counter, register set, and stack but shares the same code section and data section with other threads in the process. Threads can communicate with each other more easily since they share the same memory space.

Process: A process is an independent unit of execution that has its own memory space, resources, and program code. Each process runs independently of other processes, and they cannot directly share memory or resources with each other. Interprocess communication (IPC) mechanisms are required for processes to communicate.

6. ***How can we create daemon threads?***

Ans: *In Java, you can create daemon threads by calling the setDaemon(true) method on the Thread object before starting it. A daemon thread is a thread that runs in the background and does not prevent the Java Virtual Machine (JVM) from exiting if all non-daemon threads have finished their execution.*

Daemon threads are typically used for tasks that are not critical to the application and can be terminated when all non-daemon threads have completed. Examples include garbage collection, background logging, and other maintenance tasks.

7. ***What are the wait and cleon mashedes?***

Ans: *It seems like there might be some confusion or typo in the question regarding "wait and cleon mashedes." As of my knowledge cutoff in September 2021, there are no standard terms or concepts in computer science or programming called "wait" and "cleon mashedes." It's possible that these terms are from a specific context or a different language*