**MULTI-THREADING ASSIGNMENT**

**Q1. What do you mean by multi - threading ? Why is it important?  
Ans.** For doing a multiple task at a same time in a process , or we can say that for using a multiple apps at a same time we have required a thread for its execution and this can be done only by multithreading. That’s why we need a multithreading.  
In order to better execution of a program without any delay because as we know when we run any code, at that time single line of execution allocated for a code by default by JVM and if any delay comes in this then rest of code will not execute until upper execution will not complete. For avoiding this we required multithreading concept.

**Q2. What are the benefits of using multithreading?  
Ans.** The benefits of using multithreading are:-

* By multithreading CPU time will not goes waste.
* We can do multiple activity at a same time.
* We can access multiple apps at a same time.
* We can do multiple activities in a single app same time.
* Our execution becomes smoother.
* We can set the timings of execution also by using sleep().
* Improves the performance of CPU.

**Q3. What is thread?  
Ans.** A thread is a very light weighted process, or we can say that the smallest part of the process that allows a program to operate more efficiently by running multiple task simultaneously.

**Q4. What are the ways of implementing thread in java?  
Ans.** There are two ways of implementing threads in java are:-

* By extending thread class.
* By implementing Runnable interface.

**Q5. What is the difference between threads and process?  
Ans**. Basic difference between both of them are:-   
 **Process:-**

* Process means any program is in execution.
* The process takes more time to terminate.
* It takes more time for creation.
* It also takes more time for context switching.
* The process is less efficient in terms of communication.

**Threads:-**

* Thread means a segment of a process.
* The thread takes less time to terminate.
* It takes less time for creation.
* It takes less time for context switching.
* Thread is more efficient in terms of communication.

**Q6. How can we create daemon threads?  
Ans.** We can create daemon threads in java using the thread class setDaemon(true). It is used to mark the current thread as daemon thread or user thread. isDaemon() method is generally used to check whether the current thread is daemon or not. If the thread is a daemon, it will return true otherwise it returns false.

package Mutlithreading;  
public class DaemonThread extends Thread {  
 public DaemonThread(String name) {  
 super(name);  
 }  
  
 public void run() {  
 if (Thread.*currentThread*().isDaemon()) {  
 System.*out*.println(getName() + " is Daemon thread");  
 } else {  
 System.*out*.println(getName() + " is User thread");  
 }  
 }  
 public static void main(String[] args) {  
 DaemonThread t1 = new DaemonThread("t1");  
 DaemonThread t2 = new DaemonThread("t2");  
 DaemonThread t3 = new DaemonThread("t3");  
   
 t1.setDaemon(true);  
 t1.start();  
 t2.start();  
 t3.setDaemon(true);  
 t3.start();  
 }  
 }

**Q7. What are the wait() and sleep() methods?  
Ans.** **wait():** As the name suggests, it is a non-static method that causes the current thread to wait and go to sleep until some other threads call the notify () or notifyAll() method for the object’s monitor (lock). It simply releases the lock and is mostly used for inter-thread communication. It is defined in the object class, and should only be called from a synchronized context.

**sleep():** As the name suggests, it is a static method that pauses or stops the execution of the current thread for some specified period. It doesn’t release the lock while waiting and is mostly used to introduce pause on execution. It is defined in thread class, and no need to call from a synchronized context.