Ques 1. What do you mean by Multithreading ? Why is Important?

Ans:- Multithreading is a Java feature that allows concurrent execution of two or more parts of a program for maximum utilization of CPU. Each part of such program is called a thread. So, threads are light-weight processes within a process. Threads can be created by using two mechanisms : Extending the Thread class.

Multithreading saves time as you can perform multiple operations together. The threads are independent, so it does not block the user to perform multiple operations at the same time and also, if an exception occurs in a single thread, it does not affect other threads.

Ques 2. What are the benefits of using Multithreading?

Ans:- Some of the most important benefits of MT are:

Improved throughput. ...

Simultaneous and fully symmetric use of multiple processors for computation and I/O.

Superior application responsiveness. ...

Improved server responsiveness. ...

Minimized system resource usage. ...

Program structure simplification. ...

Better communication.

Ques 3. What is Thread in Java?

Ans:- Before introducing the thread concept, we were unable to run more than one task in parallel. It was a drawback, and to remove that drawback, Thread Concept was introduced.

A Thread is a very light-weighted process, or we can say the smallest part of the process that allows a program to operate more efficiently by running multiple tasks simultaneously.

In order to perform complicated tasks in the background, we used the Thread concept in Java. All the tasks are executed without affecting the main program. In a program or process, all the threads have their own separate path for execution, so each thread of a process is independent.

Ques 4. What are the two ways of implementing thread in Java?

Ans:- In Java, multithreading a thread can be created by the following two ways:

(a) extending the thread class.

(B)implementing a Runnable interface.

**Example:-** (a) extending the thread class.

class MultithreadingDemo extends Thread

{

public void run()

{

System.out.println("My thread is in running state.");

}

public static void main(String args[])

{

MultithreadingDemo obj=new MultithreadingDemo();

obj.start();

}

}

OutPut: My thread is in running state

(b)implementing a Runnable interface.

class Multi3 implements Runnable{

public void run(){

System.out.println("thread is running...");

}

public static void main(String args[]){

Multi3 m1=new Multi3();

Thread t1 =new Thread(m1); // Using the constructor Thread(Runnable r)

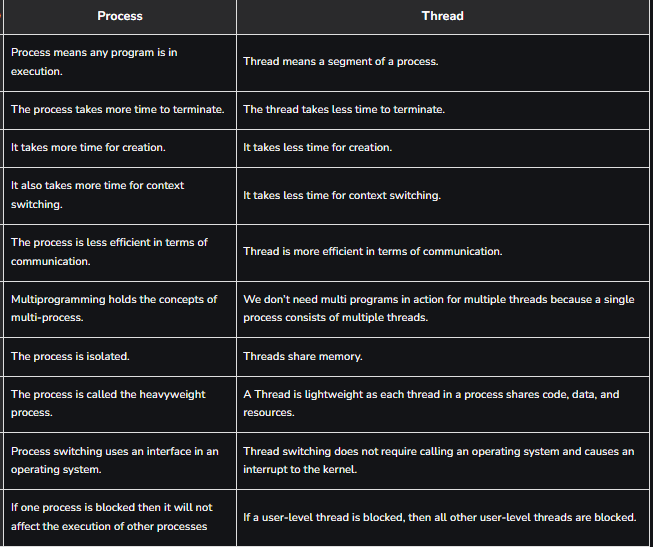
t1.start();

}

}

Ques 5. What's the difference between thread and process?

Ans:-Ther are many difference between thread and process.



Ques 6. How can we create daemon threads?

Ans:- Daemon thread in Java is a service provider thread that provides services to the user thread. Its life depend on the mercy of user threads

i.e. when all the user threads dies, JVM terminates this thread automatically.

There are many java daemon threads running automatically e.g. gc, finalizer etc.

Methods of Daemon Thread

1. void setDaemon(boolean status):

This method marks the current thread as a daemon thread or user thread. Setting a user thread as a daemon can be done using the‘tU.setDaemon(true)', while setting a daemon thread as a user thread can be done using the ‘tD.setDaemon(false)'.

2. boolean isDaemon():

This method is used to check that the current thread is a daemon. It returns true if the thread is Daemon. Else, it returns false.

// Java program to demonstrate the usage of

// setDaemon() and isDaemon() method.

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();

}

}

Output:

t1 is Daemon thread

t3 is Daemon thread

t2 is User thread

Ques 7. . What are the wait() and sleep() methods?

Ans:- Wait() in Java?

Sleep()

The Sleep () method is related to the Thread class that is used to stop the execution of the current Thread for few seconds. The Sleep () method takes the sleeping time in milliseconds. The monitor's ownership is not lost when we use the Sleep () method and start the execution again from where it stops. In simple words, the Sleep() method is responsible for sending the current Thread into the "Non Runnable" state.

Wait()

The Wait() method is related to the Object class. The Wait() method is responsible for sending the calling thread into the waiting state. The Thread remains in the waiting state until another thread doesn't invoke the notify() or notifyAll() method for that object. The Thread resumes the execution after obtaining the ownership of the monitor.

Before understanding the differences between both of them, let's understand the similarities between them. So, both the Wait() and Sleep() methods are the native methods that make the current Thread go into the Non-Runnable State.