Programming in Java

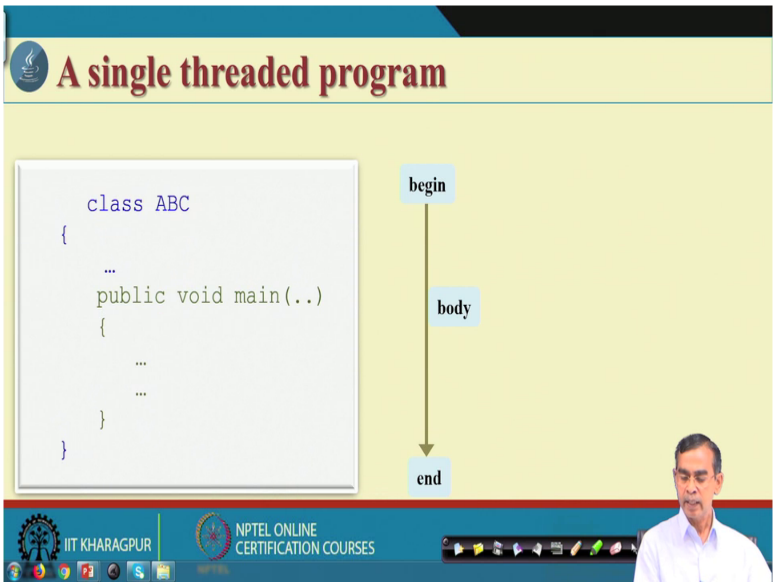
Prof. Debasis Samanta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 27
Multithreading – I

So, today we shall start about multithreaded programming very new concepts in programming arena and this multithreaded programming concept is very important to develop many different type of softwares. For example, java is best suitable for developing application software's such as library management system, word processing system like this.

It is also equally best suitable for developing system software such as compiler operating system. Now such system software development is possible using java because of this features the multithreaded programming concept. Now before going to understand about the different features so, for the multithreaded programming is concerned first we should clear our idea about what exactly a multithreading is.

(Refer Slide Time: 01:13)



A single threaded program

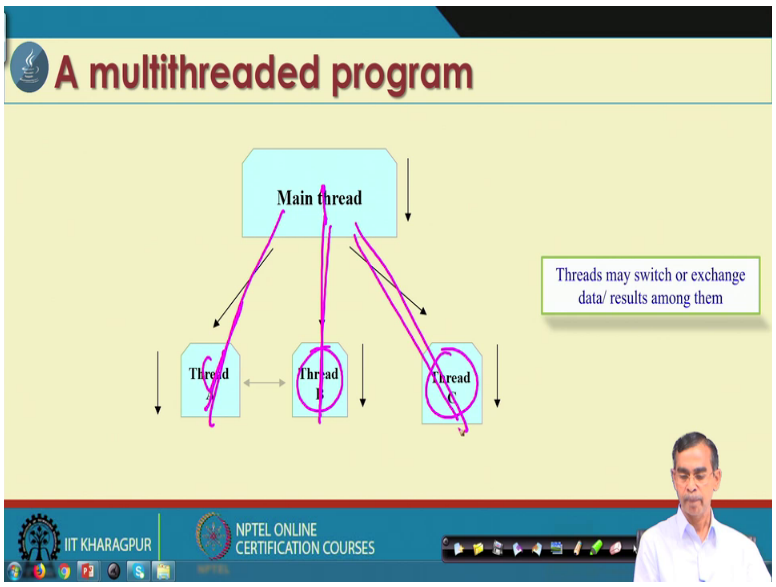
```
   class ABC
{
     ...
    public void main(..)
    {
        ...
        ...
    }
}
```

Now, as the name says the multithreading it is basically the short form of multiple thread threading actually that is why multithreading. So, definitely so, it is in contrast to single threading we can say. So, what exactly the concept of single threading is, that here we can see this is a one example of a class which has one method. So, this is a method as we see and in this method as the execution takes place from the very first statement to the next statement, the next statement and so on.

So, this way the multithreaded this way a thread will occurs. So, this way the thread will occurs starting from this to this direction. So, we can say in other way about that if we start from with the first statement like so is a begin the next statement, the next statement, next statement and so on and finally, it will go to the end of the statement. And this way the threads or the execution that it takes place is called an a single threaded manner and then the execution is called the single threaded execution.

Now, in contrast to this single threaded execution there is a concept called multi thread execution.

(Refer Slide Time: 02:51)



A multithreaded program

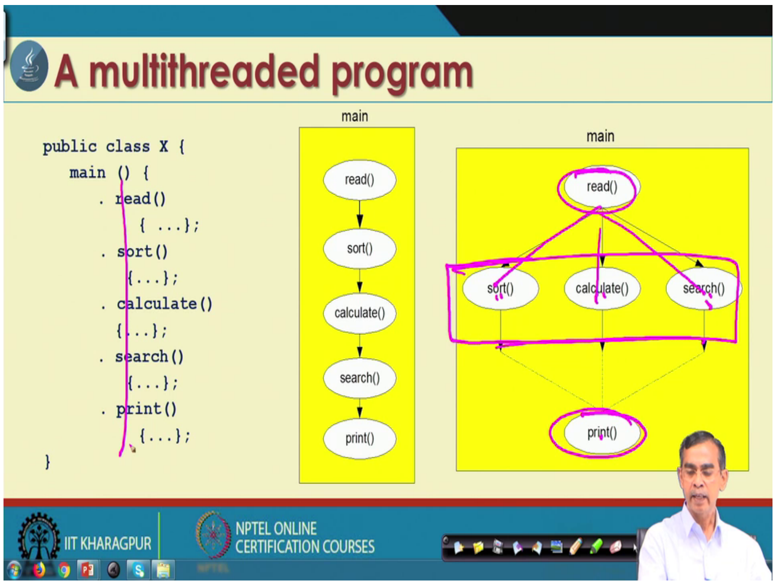
Threads may switch or exchange data/ results among them

Here is an example that we can use this example to explain the concept of multiple threading here. So, execution here will start from this point suppose and then if we can bifurcate this execution into 3 or more here for example, bifurcation into this is a one part, this another part and this another part then it is called that. So, the execution will takes place in a concurrent manner; that means, it will run this part, it will run this part and it will run this part. So, all 3 executions can takes place in a parallel manner or in a concurrent manner then we can say this is a multi threaded execution.

So, this is a idea about the multithreading now here one important point that we should note here is that whenever the threads are in execution all this threads can exchange the data among themselves. Here for example, the result that is produced by these thread, can be used by these thread or vice versa like this, also they can use a global data among them also not global data means common data among them also anyway. So, these thread whenever they execute, they execute in parallel and then the data exchange is quite possible among them

Now, as an another example we can consider one more another example.

(Refer Slide Time: 04:21)



A multithreaded program

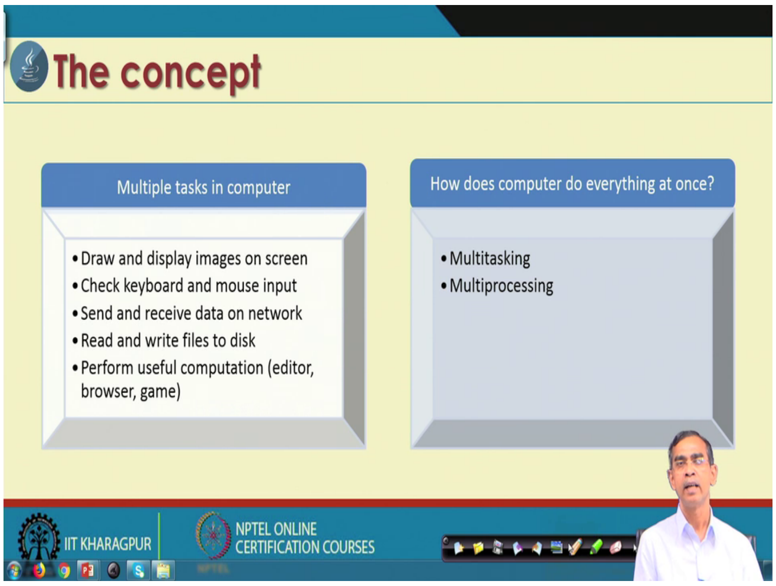
```
public class X {
    main () {
       . read()
            { ...};
       . sort()
           {...};
       . calculate()
          {...};
       . search()
           {...};
       . print()
            {...};
}
```

Let us consider this is a one part of the program and here as we see this program or you can say the main method consist of different modules. So, this is a one module, this another module, search module, clean module so, different modules are there. Now if this program executes in a single threaded manner this means that this module will start it is execution completes then only the next module start and so on. So, this way the single threaded execution will takes place. So, pictorially the single threaded execution will look like this pictorially the single threaded execution will look like this so, it is this way.

Now on the other hand if the same thing which if we run in a multiple threaded then it will look like this. So, if we run in a multiple threaded then as we see here. So, in a multiple threaded as you see. So, after the reading operation is there it will bifurcate into

3 different what is called the parts, the sort, calculate and search. So, what you can say at this stage there are 3 threads, sort thread, calculate thread and search thread. They can execute in parallel and once all this threads are finished their execution it will come to this thread and then print will execute. So, this way this parts can execute in a multiple threaded manner, this is single threaded manner. So, this is a concept of multithreaded execution or is a program.

(Refer Slide Time: 06:33)


The concept

Multiple tasks in computer

- Draw and display images on screen
- Check keyboard and mouse input
- Send and receive data on network
- Read and write files to disk
- Perform useful computation (editor, browser, game)

How does computer do everything at once?

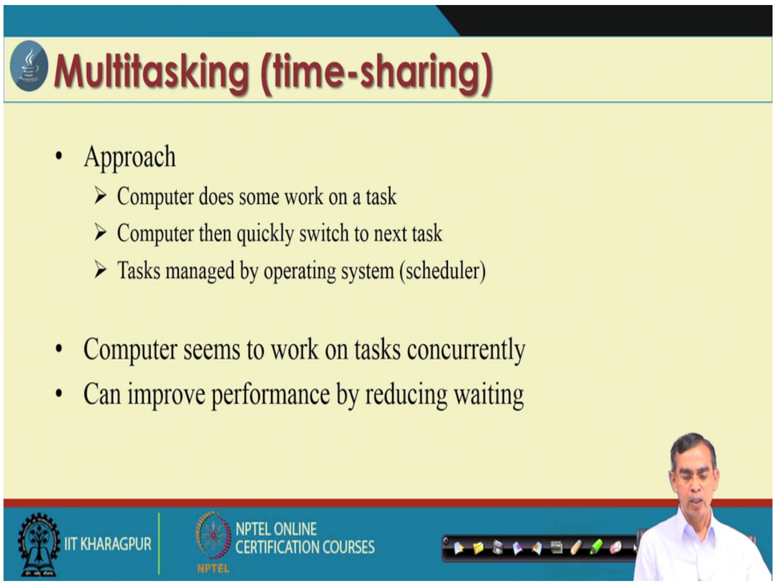
- Multitasking
- Multiprocessing

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, now, we have learnt about the idea about the multithreading how it works is there, there are many example that can be given why are the multithreaded execution is obvious. So, whenever we are using our system computer there are many task which is basically works in a continuous concurrent manner. For example, when you are reading web page at the same time you can type into using word processor and at the same time you can play your music. So, all the things are going in parallel. So, for everything for playing a music there is a one thread for browsing the information from the internet one thread whenever you are typing something using a word processor application is another thread and all these things.

Now so there are many applications where the multithreaded way the executions are there and to be carried out. So, that we can achieve maximum things from the system, now how this multithreading actually can be achieved in our system, there are in fact, 2 ways the multitasking and multiprocessing. So, in our few next slide we should understand about what is the multithreading and multitasking is.

(Refer Slide Time: 07:53)



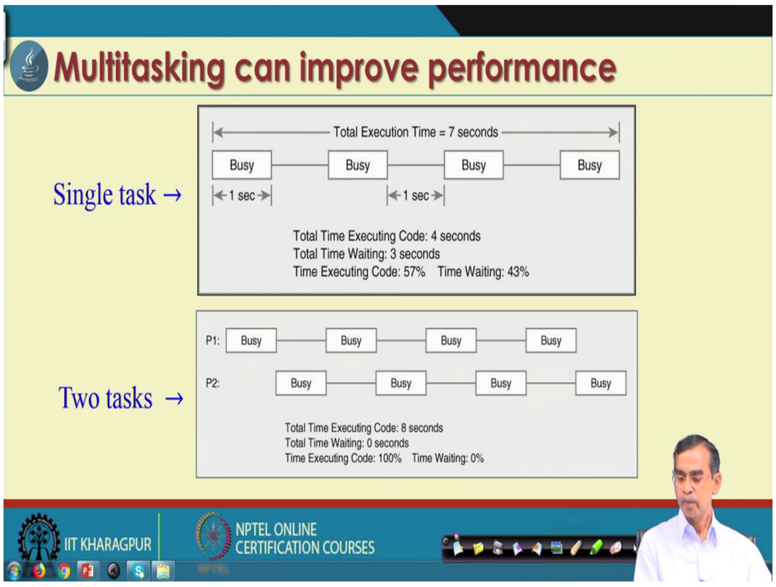
Multitasking (time-sharing)

- Approach
  - Computer does some work on a task
  - Computer then quickly switch to next task
  - Tasks managed by operating system (scheduler)
- Computer seems to work on tasks concurrently
- Can improve performance by reducing waiting

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
NPTEL

So, here the multitasking is also alternatively called the time sharing; that means, if we have to run the multiple programs together. So, is a part of one program can share the CPU time or some other resource time and at the same time whenever the CPU is busy, it can share some other resource and this one or actually the idea is that multiple programs have their own time slots and each program can use their time slots so, that they can execute this program. So, this is the way the multi tasking things can be achieved and in a time sharing manner.

(Refer Slide Time: 08:42)


Multitasking can improve performance

Single task →

Total Execution Time = 7 seconds

Busy — Busy — Busy — Busy

1 sec   1 sec

Total Time Executing Code: 4 seconds
Total Time Waiting: 3 seconds
Time Executing Code: 57%   Time Waiting: 43%

Two tasks →

P1: Busy — Busy — Busy — Busy

P2: Busy — Busy — Busy — Busy

Total Time Executing Code: 8 seconds
Total Time Waiting: 0 seconds
Time Executing Code: 100%   Time Waiting: 0%

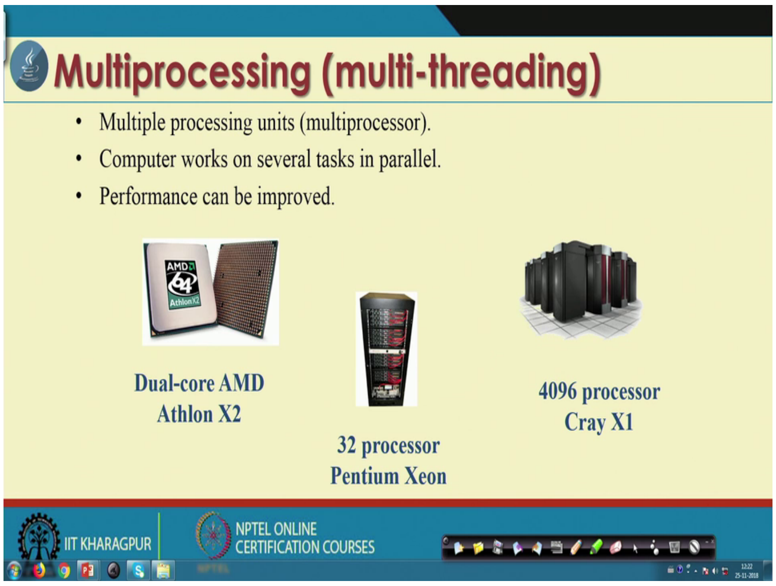
Here is an example that we can use to explain how the multitasking it is possible and. So, say is a basically single task if it is a single task as we see whenever the single task require the CPU time it is their and if it requires at the same time some other resources say suppose input then the CPU is idle. On the other hand once the input is reading is complete then the CPU is engaged for some other task one this task is complete then it may be some other task say some printing and one printing is over another task and so on so on.

So, what we can see is that CPU is not busy for all the time whenever it is executing a program if it is a single threaded because of some other resource requirements. So, this way actually the a performance or efficiency or the throughput of the system is less as we can see if we execute this way the code is executed only 57 percent where the system remain idle for 43 percent. So, this is not an efficient way of solving this program.

On the other hand if we run 2 programs concurrently say P 1 and P 2 for example, whenever CPU is idle, at the same time if we can engage the CPU by other program then whatever the ideal time can be vanished and then CPU can utilize 100 percent utilization. So, this way the multitasking if it is possible here actually we can give the time sliced P 1 into this time then next this time and this time for the P 1 this time for P 2 and so on. So, the time can be sliced among the processes. So, this way it is not only for the 2 processes even multiple processes n number of processes can be time shared and time slice can be given to each processes. So, that CPU will be busy and then the program all programs will be executed in the fastest way.

Now, so, this is the idea about the multitasking is a one way of doing multiple threading actually.

(Refer Slide Time: 10:56)



Multiprocessing (multi-threading)

- Multiple processing units (multiprocessor).
- Computer works on several tasks in parallel.
- Performance can be improved.

Dual-core AMD Athlon X2

32 processor Pentium Xeon

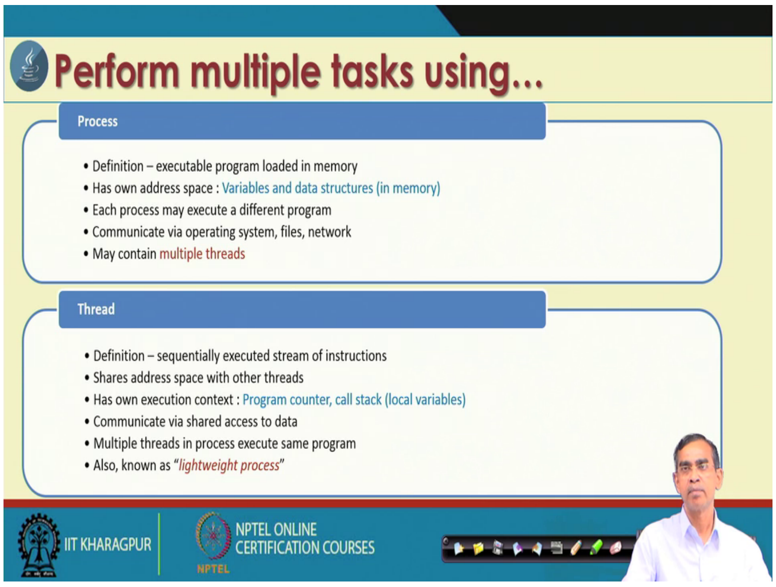
4096 processor Cray X1

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

So, now other than this the multitasking there is one called the multiprocessing. So, it is basically the idea is same it is also multithreading, but the pro idea is called multiple processes can be engaged here. So, if there are say modules m 1 to m n so, all this modules can be assigned to different processors. So, that all the processors can run concurrently in a parallel manner.

So, this way the program the entire program can be executed in a faster way and there are many multiprocessing systems are available today, as we know for example, here Pentium Xeon having the 32 processors you need Athlon X 2 MD is a dual core processor and then there are 4096 processor Cray X1 having so many processor now. So, these are the high speed competing environment which basically allows to execute a program in a multi threaded manner.

(Refer Slide Time: 12:05)



Perform multiple tasks using...

Process

- Definition – executable program loaded in memory
- Has own address space : Variables and data structures (in memory)
- Each process may execute a different program
- Communicate via operating system, files, network
- May contain multiple threads

Thread

- Definition – sequentially executed stream of instructions
- Shares address space with other threads
- Has own execution context : Program counter, call stack (local variables)
- Communicate via shared access to data
- Multiple threads in process execute same program
- Also, known as "*lightweight process*"

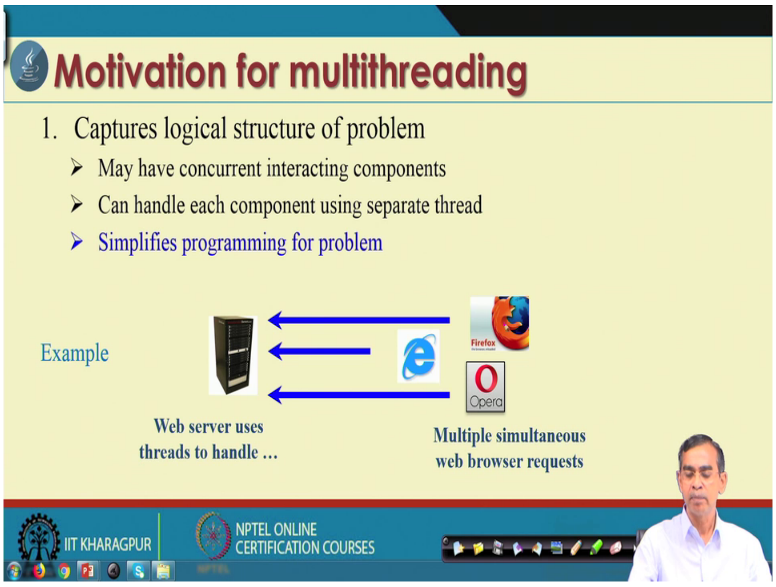
Now, so, here is basically the 2 concepts. So, for the multitasking is concerns so, process and thread. So, basically the entire programs can have the different parts this basically the process. So, process is basically is a program is an executable form which can be loaded into the memory and then once the processors are loaded into the memory operating system can take care the execution of each processes.

And on the other hand there is a concept of threads which basically the idea about that how a process can execute in the system. So, it is basically it is a thread is basically a sequential execution of a set of instructions one by one. So, there are multiple thread means there is a multiple set of sequential execution of a systems which can share the data among them.

So, multiprocessors multiple processes this basically multiple threads we can say and whenever one thread come into the picture it is basically is a single process with a single set of execution. So, multiple threads is basically a concurrent execution whereas, a single thread execution is called the light weight process because it is a on thread actually.

Now, so, we have learn about the multithreading and we can learn that a multithreading means a concurrent execution. Now let us see how this multithreading is useful to solve many problems as we are solving nowadays in our own computing system.

(Refer Slide Time: 13:52)



Motivation for multithreading

1. Captures logical structure of problem
   - May have concurrent interacting components
   - Can handle each component using separate thread
   - Simplifies programming for problem

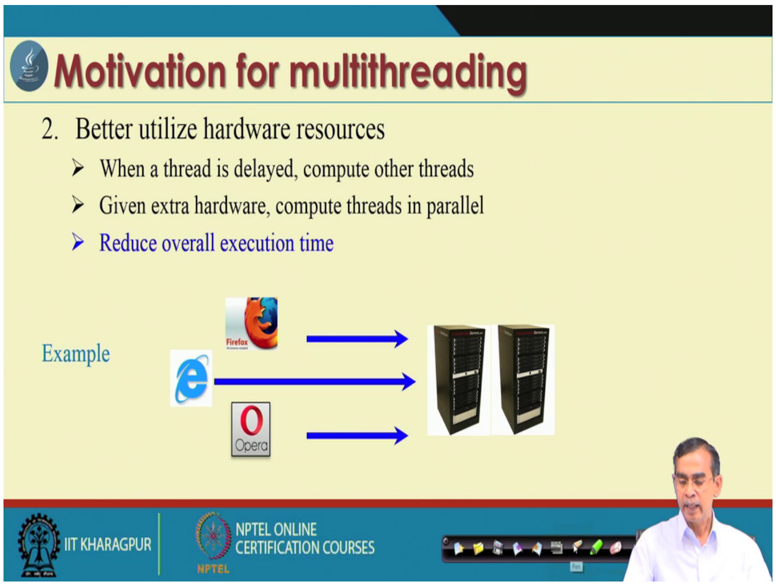
Example

Web server uses threads to handle …
Multiple simultaneous web browser requests
Firefox
Opera
IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

Now, you know whenever you are using the Gmail software or a Google browser. So, Gmail software is basically loaded into some server is a Google server Gmail server and you know at the same time the billion of peoples are using the same Gmail server. In fact, then how this Gmail server processes all the Gmail users actually at the same time. So, it is basically they the Google server which basically run the Gmail software's in their own system they processes all the users in a multiple threaded manner so, this is a one example of the multiple threading.

As we see from here a user a user can place a request or can browse the Gmail from the say Firefox, another Internet Explorer, another may be Opera so all the request goes to the server at the same time so, billion of requests are there, then this server can process them all the request individual in a multithreaded program. However, a user can think that this server is dedicated to his service. So, all user can think that I am getting as if the I am the only soul user in the system, actually the Google server serves multiple users at the same time and this is possible because of that multithreading concept as it is followed in execution.

(Refer Slide Time: 15:26)



Motivation for multithreading

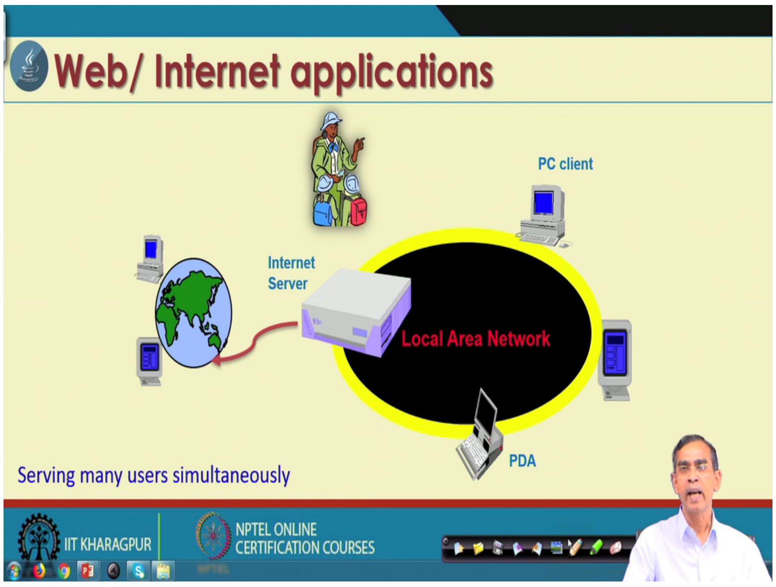
2. Better utilize hardware resources
   - When a thread is delayed, compute other threads
   - Given extra hardware, compute threads in parallel
   - Reduce overall execution time

Example

Firefox
Opera
IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
NPTEL

Now, so, there is idea about this one and then basic purpose of this multi and this is another example another example of multithreading using multiple processor. So, idea it is that if so, many users one single server cannot handle then at the end the multiple server can take care and then they can use it. So, here only 2 server as you see instead of 2 server a large number of servers can take care and then multithreading also can be done in a multi processing manner also. So, basic idea about that it will not only process multiple programs in execution, but it reduces overall time of executions for all programs. So, this is a great advantage that one can achieve using multithreading manner.

(Refer Slide Time: 16:18)



Web/ Internet applications

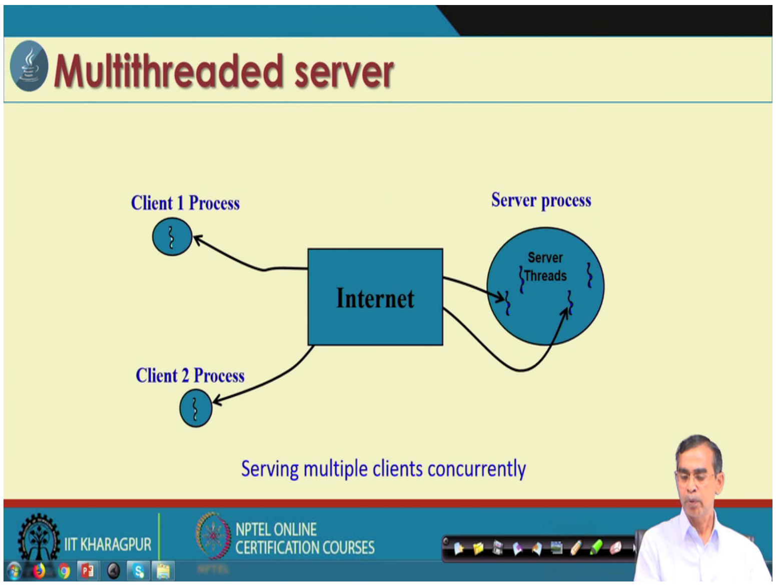
Serving many users simultaneously

And this is another example as we see in the context of internet programming. So, the idea is that if a many users are there in network access the network with different devices like laptop, mobile phone, PDA, PC personal computer whatever it is there now all the users those are there in the system they are connected to a server called the proxy server and then.

So, these proxy server can know that which request to first 2 home in what way so, if so, there is basically the different other servers those are distributed worldwide and then from this internet server the request will go and then, they basically possess all the requested and that again that are the end of the service.

So, another request also come from another network they also process and then they produce the output and the output go to the again this proxy and then they can again transmit to this way and this way the multiple execution takes place. So, for the internet programs are concerns. So, this is a concept of internet programming and internet programming is today possible because of the concept of multithreading as it is possible.

(Refer Slide Time: 17:47)

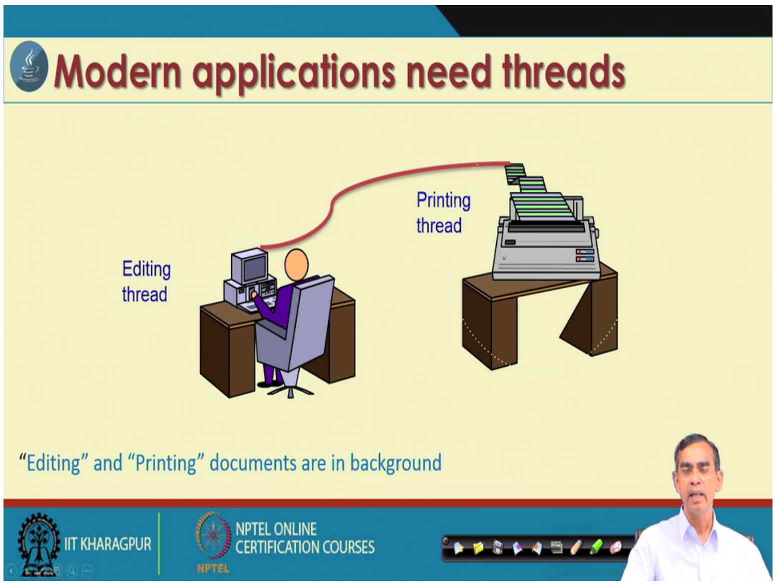


Multithreaded server

Serving multiple clients concurrently

So, there are multiple servers that can execute in parallel as we see here for example, these are internet server and there is basically service request that receives why a server that needs for to process the different clients and then here as we see this server can run many threads. So, they are called parallely. So, overall time that is required to get it serviced will be reduced remarkably and that way we can improve the performance of the system.

(Refer Slide Time: 18:27)



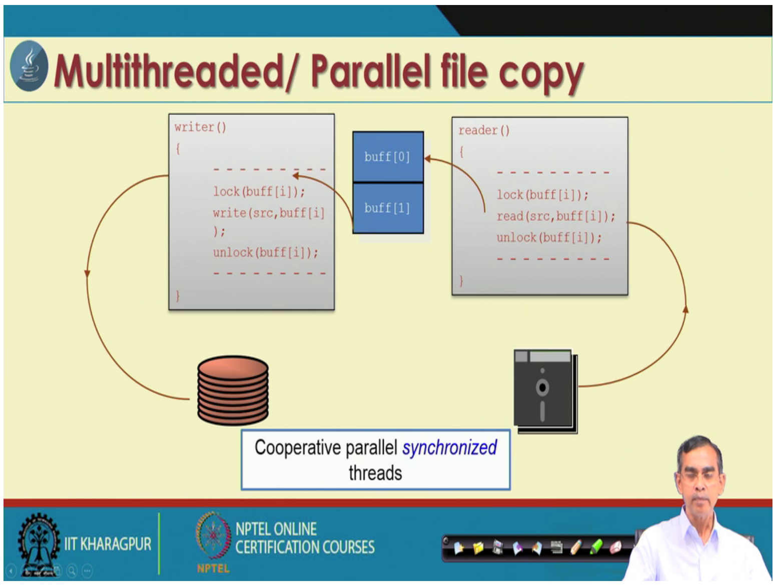
Modern applications need threads

"Editing" and "Printing" documents are in background

So, there are many example as we see where the multithreading is there now this is the in the context of internet related it application now even in our single PC system also we need the multiple threading. Here for example, the user whenever editing a document at the same time he can fire a print come on to print one paper. So, to him both the things editing as well as printing takes place together. So, this is also possible because here the system can take care 2 threads one editing thread another is a printing thread. So, is a pages will be printed one by one it will control the printer buffer and then get the response and then do it. So, the system which is here can control both the applications together.

So, this is one example even in the single user in moment also the multithreading is required in our problem solving.

(Refer Slide Time: 19:26)



Multithreaded/ Parallel file copy

```
writer()
{
    - - - - - - - - - -
    lock(buff[i]);
    write(src,buff[i]
    );
    unlock(buff[i]);
    - - - - - - - - - -
}
```

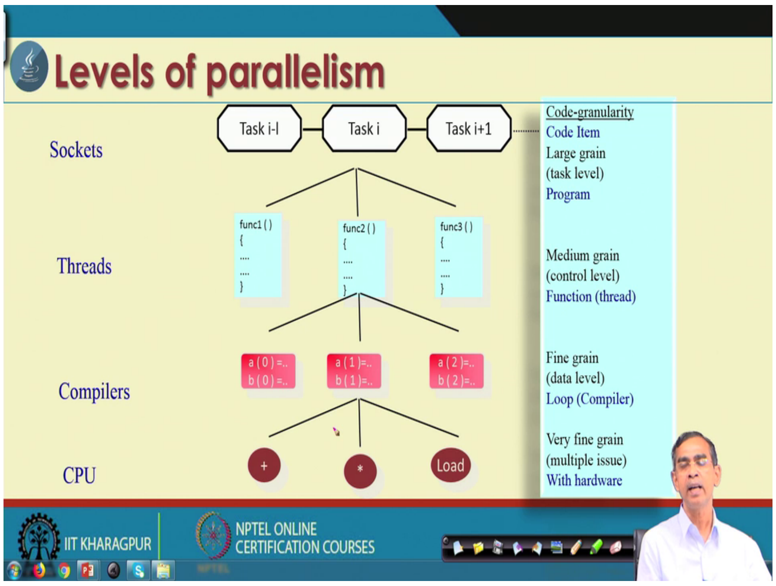
```
reader()
{
    - - - - - - - - - -
    lock(buff[i]);
    read(src,buff[i]);
    unlock(buff[i]);
    - - - - - - - - - -
}
```

Cooperative parallel *synchronized* threads

And here is another example also. So, for the operating system level is concerned we also require the multithreading say suppose we are reading something from our hard disk at the same time we are copying something from our pen drive or floppy disk to hard disk. So, how this thing is possible again this is possible because of the multithreaded program. So, they are actually 2 programs one is a writer and the reader, the reader will read from the floppy hard disk and writer will write in to the hard disk and both things will takes place together in a multiple threaded manner

So, this thread and this thread will execute in parallel and in a multithreaded manner actually. So, there are several applications as we see where the multithreading is must ok. So, we have learned about why multithreading now this is our turn to learn about how this multithreading is possible in java. There are many features many concepts are know available in this context who you will learn all this features one by one in the next few slides.

(Refer Slide Time: 21:02)



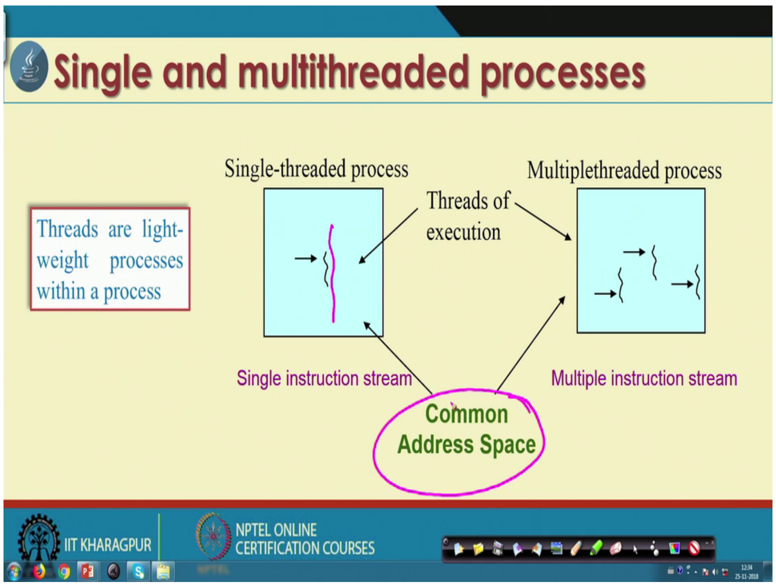
So, here is basically the multithreading is done in the different level, first is the code level, then medium grain the control level and then fine grain it is a data level and these way very fine grain the data level it is there. So, here basically in the first code level multithreading is possible because we have to first decompose the program into different task where all task can be done independently.

So, at the task level or is a code level is called the code granularity or the task level the multithreading will start first; that means, you have to when you write the program you can tell that this is the piece of the code and this is other piece of the code the 2 task they can do independently and then 2 threads can be planned for that.

One it is there then we can write the threads for each and then the all the threads can be takes place and then at the execution level it basically creates the process for each thread is basically the code that is executable for example, in case of your java execution the bite codes for each threads actually it is there. So, it is a so, for each thread there is a separate codes that can be placed there and finally, at the CPU level for each code can takes place in a inter living or in a concurrent manner as it is basically, whether that is a operating system can manage how this task, this task and this task can be carried out parallely. So, they can use some process synchronization from time sharing or many other methods that the operating system can follow for you to execute to more than one thread together.

So, this is the idea about the parallelism is takes place in this right and java programming is basically helps us to implement all this concept here, but at it is a very high level concept. So, we do not have to bother. So, much details which is basically there only we have to think about this code level and then, this is a that code level if we can plan it; that means, program level we can say we if we can plan it, then we will be able to achieve the multithreading. So, java exactly does for these things for us for the programmer.

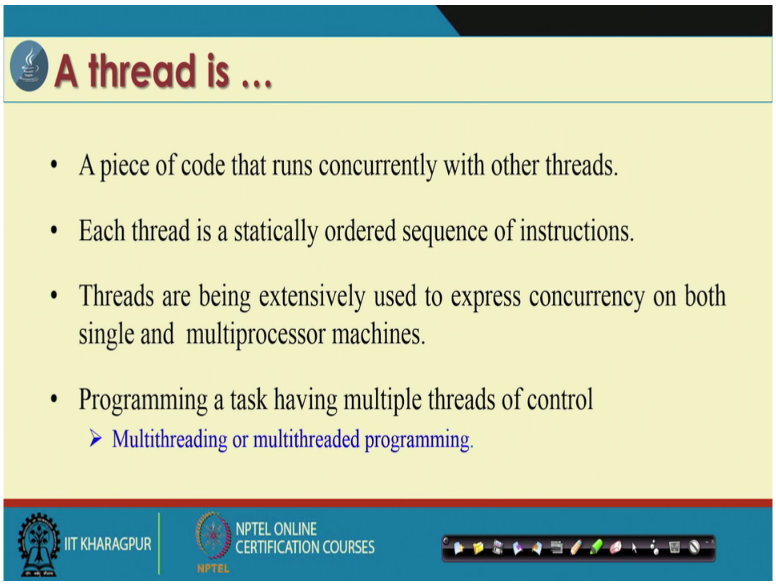
(Refer Slide Time: 23:37)


Single and multithreaded processes

Threads are light-weight processes within a process

Now, so here basically single thread verses multiple thread as we say earlier that this is the one single thread if it is there one execution, if the multiple thread mean multiple executions and they can use the common memory common data together actually this is the idea.

(Refer Slide Time: 23:59)



A thread is ...

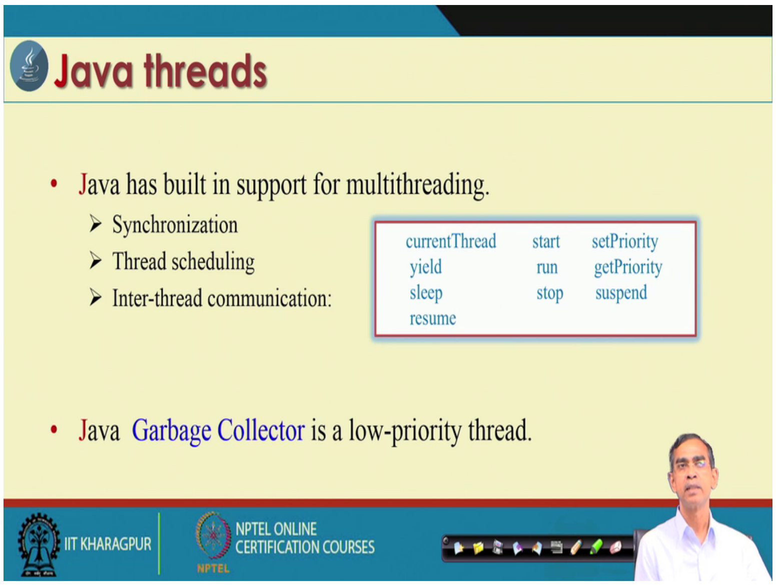
- A piece of code that runs concurrently with other threads.
- Each thread is a statically ordered sequence of instructions.
- Threads are being extensively used to express concurrency on both single and multiprocessor machines.
- Programming a task having multiple threads of control
  - Multithreading or multithreaded programming.

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
NPTEL

Now, a therefore, what we learnt is as a summary that, a thread is a piece of code that runs in a parallel manner with other threads and each thread is a statically ordered sequence of instruction; that means, each thread is basically one set of a sequential instruction we can say and all threads are extensively used to express concurrency whether in a single or multiprocessor machine whatever it is there that mean either in a time sharing or in a multi processing whatever it is there.

Now, let us see the features which are there so, for the multithreading is concern.

(Refer Slide Time: 24:38)



Java threads

- Java has built in support for multithreading.
  - Synchronization
  - Thread scheduling
  - Inter-thread communication:

currentThread	start	setPriority
yield	run	getPriority
sleep	stop	suspend
resume		

- Java Garbage Collector is a low-priority thread.

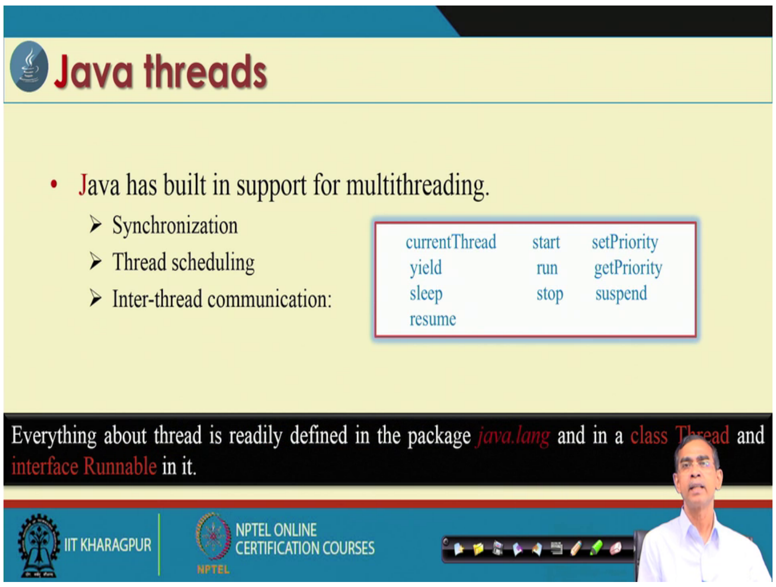
IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES
NPTEL

Mainly 3 things that we have to learn about that, how to create a thread and how to schedule the thread so, these are first thing and then second thing is that how the inter thread communication can be takes place and finally, the synchronization of this threads. So, in java there are many what is called the methods are available and there is again package the package which basically responsible for running multiple threads are there.

Now the different process that is related to I mean all this task like thread creation, thread scheduling, thread execution and then inter process, thread communication synchronization there are few methods which we have mentioned here these are the methods is basically there.

Now these are the threads execution is done by is basically automatic one manager. So, it is run time manager we can say the thread manager and this automatically run for you now an example of a thread which usually occur in whenever you run a java program is called the garbage collector there is also. So, this garbage collector is basically take the memory and then how the memory can be efficiently used.

(Refer Slide Time: 26:00)


Java threads

- Java has built in support for multithreading.
  - Synchronization
  - Thread scheduling
  - Inter-thread communication:

currentThread	start	setPriority
yield	run	getPriority
sleep	stop	suspend
resume		

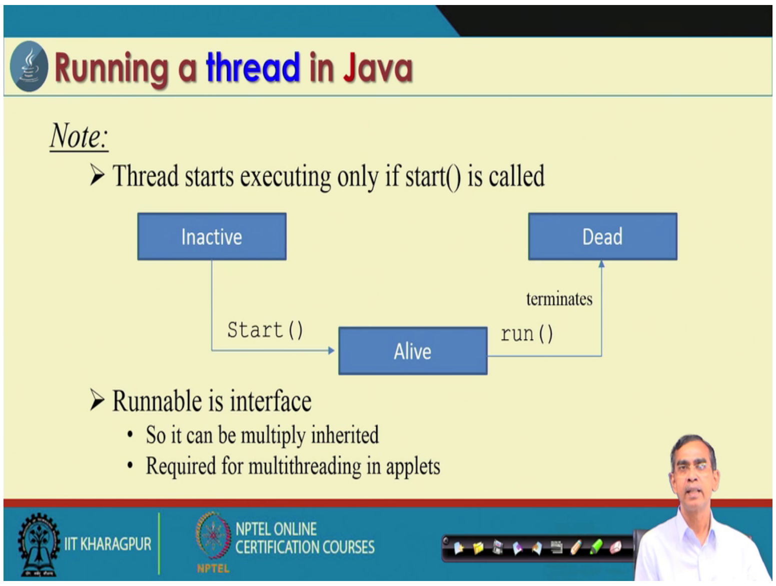
Everything about thread is readily defined in the package *java.lang* and in a class Thread and interface Runnable in it.

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

Now, so, there are many other concepts are there which basically to be considered whenever we have to schedule the thread and then control inter process communication among the different threads and then synchronizations.

Now, for all this activities there is a package which is there in the java jdk called the java dot lang and in the java dot lang there is a class thread and one interface called Runnable. So, these are the 2 main things that is there in the java and so, for the multithreading learning is concerned we have to learn this class thread and interface runnable and using this thread and interface runnable we have to learn about how the multithreading is possible.

(Refer Slide Time: 26:46)



Running a thread in Java

Note:

- Thread starts executing only if start() is called

Inactive
Dead
terminates
Start()
run()
Alive

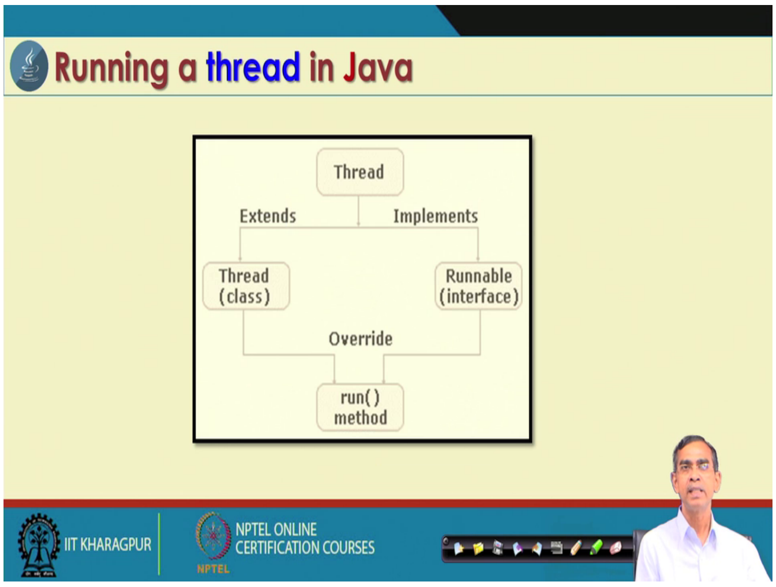
- Runnable is interface
  - So it can be multiply inherited
  - Required for multithreading in applets

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

Now, the basic concept about so, for the creating a thread is concerned you write a program for a thread that is basically the task and then once the thread is ready then you have to first run the thread and then before running we have to start this. So, basically it start and then this so, whenever a thread is build it is basically inactive thread and if you can start it the thread become a alive and from the alive threads run until it terminates when the thread is terminated the thread is called the dead.

So, doing I mean for controlling, all this start run and everything. So, there is a procedure which is defined methods which are defines there in the class thread and in the runnable interface. So, as you know it is an interface this means the methods are abstract and public that mean we have to write our own code how to run the thread and obviously, how to start it.

(Refer Slide Time: 27:44)

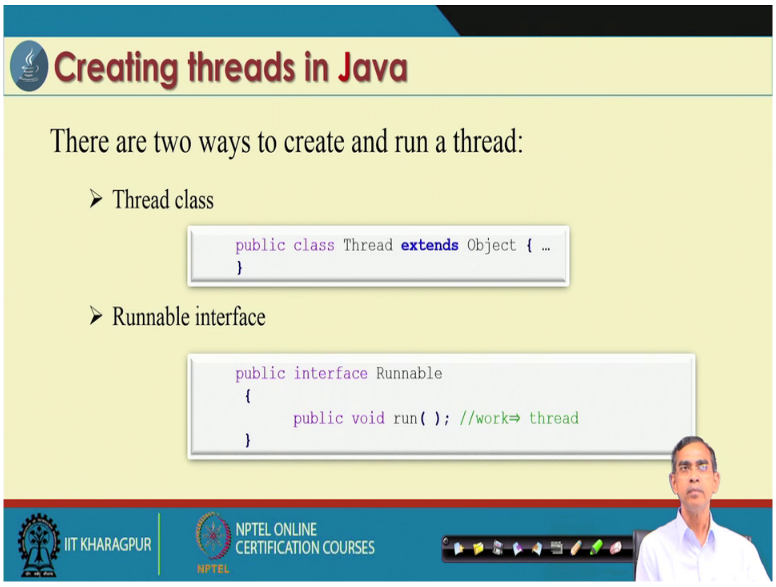


Running a thread in Java

Now, basically the main idea about that if we use a thread as a class which is there and then your thread should inherit these thread class to build the run method. So, it is basically we have to override the run method if we use thread class. On the other hand if we use the runnable interface, then we have to implement the interface using your thread class and then they are also you have to implement the run method. So, this is the basic concepts.

Now, we will discuss about using the both how the thread can be created and then how the thread can be executed. So, let us first start the discussion about how we can create a thread with the thread class.

(Refer Slide Time: 28:25)



Creating threads in Java

There are two ways to create and run a thread:

- Thread class

```
public class Thread extends Object { …
}
```

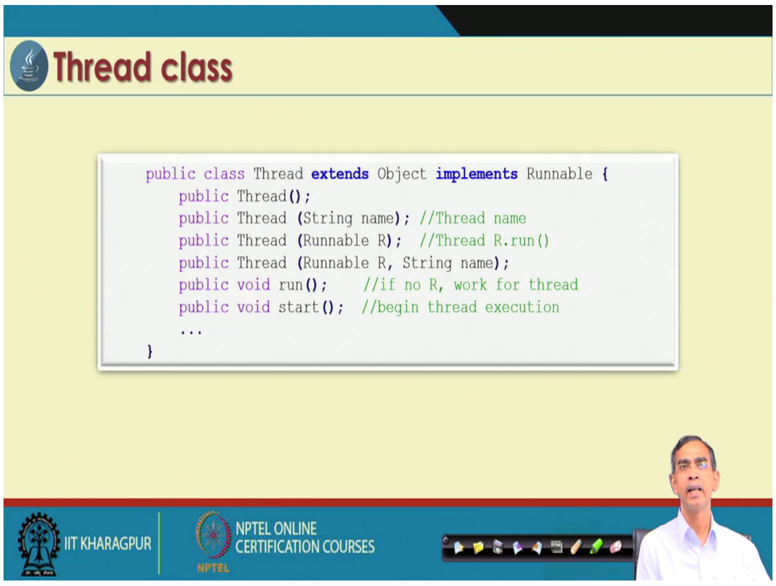
- Runnable interface

```
public interface Runnable
 {
        public void run( ); //work⇒ thread
 }
```

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, idea is very simple we have to extends the java dot lang package thread class and a thread class can be created.

(Refer Slide Time: 28:39)



Thread class

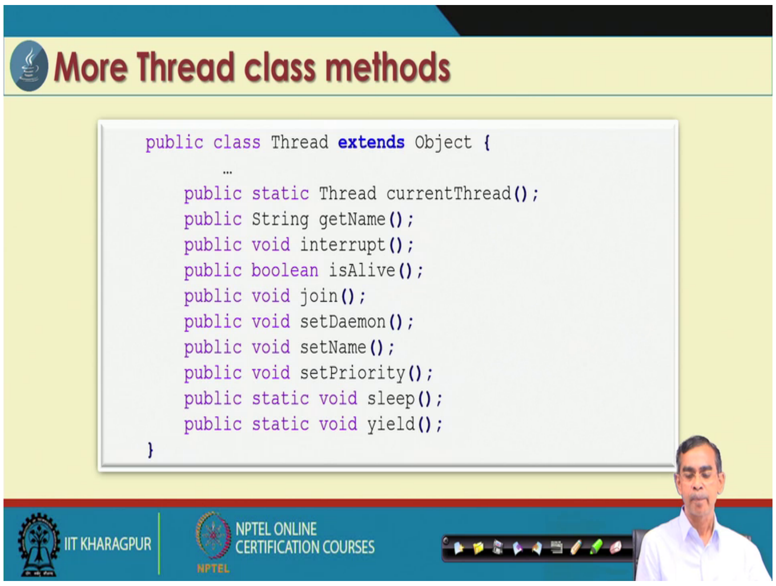
```
public class Thread extends Object implements Runnable {
    public Thread();
    public Thread (String name); //Thread name
    public Thread (Runnable R);  //Thread R.run()
    public Thread (Runnable R, String name);
    public void run();     //if no R, work for thread
    public void start();  //begin thread execution
    ...
}
```

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

And if you use a interface then we have to implement the interface here now here is an example that we can see how we can create a thread and what are the methods are there. So, here basic idea about that this is the class thread extends objects implements runnable whatever it is there. So, public thread these are the methods these are the constructor I can say the thread has the constructor in a different form where name of the thread the interface runnable object can be pass to them and then both name and threads and everything and they are run into. So, these are the basic structure of the class thread which is defined in the java dot lang package. So, it is simple not so, very complex things are there, now we will slowly one learn one by one how all those things can be handled in our program.

So, this is about the thread class as we see there are constructors and there is a 2 methods run and start and all this 2 methods are as it is basically override because it is abstract whenever it is there.

(Refer Slide Time: 29:42)

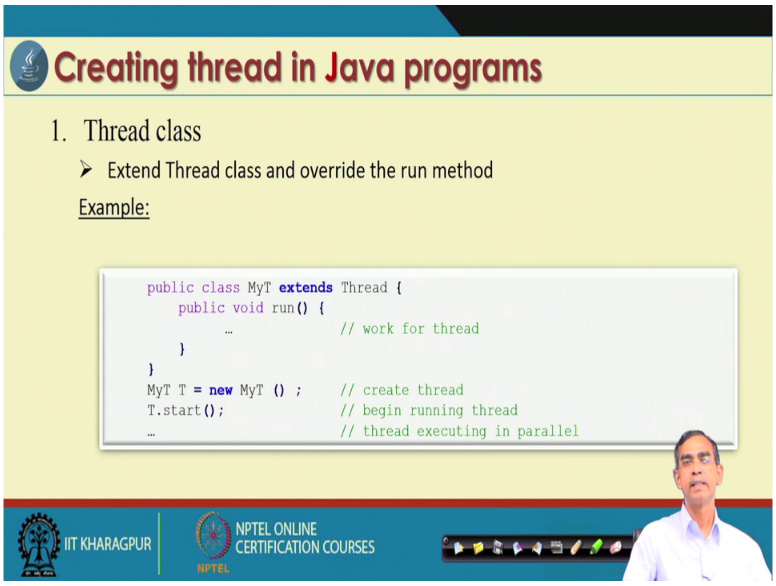


More Thread class methods

```
public class Thread extends Object {
        …
    public static Thread currentThread();
    public String getName();
    public void interrupt();
    public boolean isAlive();
    public void join();
    public void setDaemon();
    public void setName();
    public void setPriority();
    public static void sleep();
    public static void yield();
}
```

And so, for the different methods are concerned in the thread class other than the this start and run method also it includes few more methods which we have listed here join get live, get name, is alive, setting priority, sleep, yield and so, on. So, these are the few methods are very learning all this methods will basically completes our learning about the thread execution.

(Refer Slide Time: 30:25)



Creating thread in Java programs

1. Thread class
   - Extend Thread class and override the run method

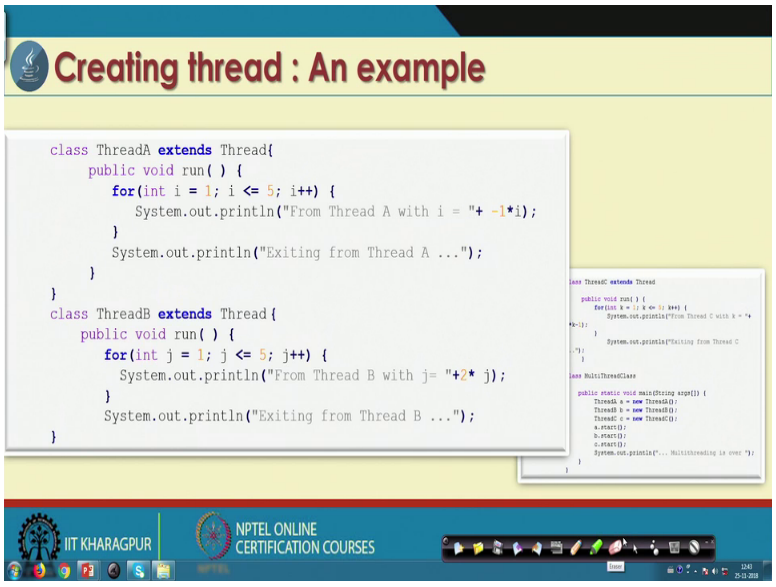
Example:

```
public class MyT extends Thread {
    public void run() {
          …              // work for thread
    }
}
MyT T = new MyT () ;      // create thread
T.start();                // begin running thread
…                         // thread executing in parallel
```

Now let us first discuss about how we can create our own thread using class thread. So, this is basically typical steps or procedure that you should follow in your program. So, here you have to declare a class of your own that is basically your thread class which should extends the thread class that is there in java dot lang package and then these are run method we have to override this run method in your program and then once this thread class is created you can use this thread class to create your own thread.

So, this is your own thread and this is basically creating a thread objects and then if we call the start method for this thread object. So, the thread will start it is execution and then the run will be invoked. So, the run method will be invoked so, that the alive thread can execute in the system.

(Refer Slide Time: 31:28)



Creating thread : An example

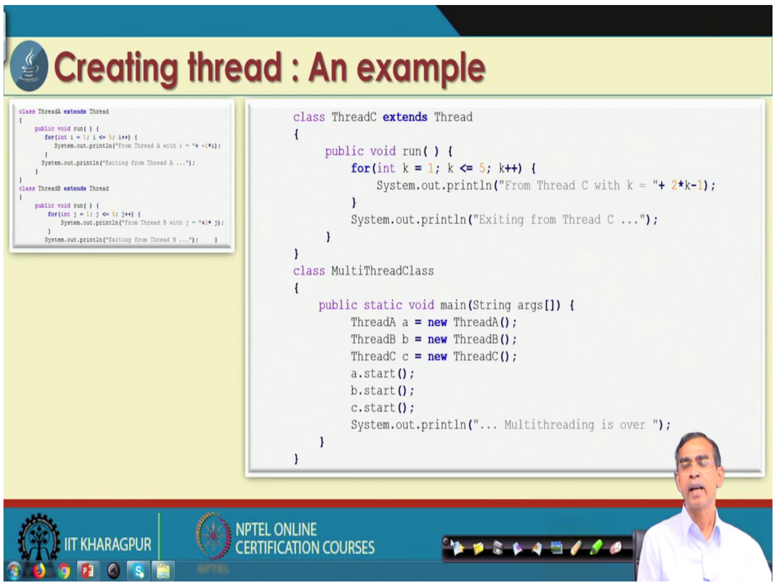
```
class ThreadA extends Thread{
      public void run( ) {
         for(int i = 1; i <= 5; i++) {
            System.out.println("From Thread A with i = "+ -1*i);
         }
         System.out.println("Exiting from Thread A ...");
      }
}
class ThreadB extends Thread {
     public void run( ) {
        for(int j = 1; j <= 5; j++) {
          System.out.println("From Thread B with j= "+2* j);
        }
        System.out.println("Exiting from Thread B ...");
}
```

```
lass ThreadC extends Thread

   public void run( ) {
       for(int k = 1; k <= 5; k++) {
           System.out.println("From Thread C with k = "+
*k-1);
       }
           System.out.println("Exiting from Thread C
..");
   }

lass MultiThreadClass

  public static void main(String args[]) {
       ThreadA a = new ThreadA();
       ThreadB b = new ThreadB();
       ThreadC c = new ThreadC();
       a.start();
       b.start();
       c.start();
       System.out.println("... Multithreading is over ");
  }
}
```

Now, here is basically the idea about idea about creating the program this is a complete program that we can see here and as we see this is the one thread class as we have declared. So, class thread A extends this so, thread A we create a thread of our own and this thread, we in this thread we declare the run method of our own you can see the run method is declared here it is basically there is a loop, loop will roll for i equals to 1 to i less than equals to 5 and whenever loop will roll it will print this 1. So, what we can say that these thread; thread A whenever it will run it will print then 5 negative numbers one by one.

On the other hand there is another thread we have created thread B, it is similar to this thread, but here in this run method we define this kind of code it basically run the even numbers. So, 2 threads as we see we have declared here and once the threads are declared we can use it. So, these are the 2 threads in addition to these 2 threads there is one more threads also.

(Refer Slide Time: 32:53)



Creating thread : An example

```
class ThreadA extends Thread
{
     public void run( ) {
        for(int i = 1; i <= 5; i++) {
           System.out.println("From Thread A with i = "+ -1*i);
        }
       System.out.println("Exiting from Thread A ...");
     }
}
class ThreadB extends Thread
{
     public void run( ) {
         for(int j = 1; j <= 5; j++) {
             System.out.println("From Thread B with j = "+2* j);
         }
        System.out.println("Exiting from Thread B ...");    }
```

```
class ThreadC extends Thread
{
     public void run( ) {
         for(int k = 1; k <= 5; k++) {
             System.out.println("From Thread C with k = "+ 2*k-1);
         }
         System.out.println("Exiting from Thread C ...");
     }
}
class MultiThreadClass
{
    public static void main(String args[]) {
         ThreadA a = new ThreadA();
         ThreadB b = new ThreadB();
         ThreadC c = new ThreadC();
         a.start();
         b.start();
         c.start();
         System.out.println("... Multithreading is over ");
    }
}
```

So, thread A and thread B that we have discussed in addition to thread and there is another thread C these basically also the similar run method which we have discussed here. So, what we can say that we have created 3 threads thread A, thread B, thread C, creating 3 thread is basically the business about that how we can define the run method; that means, the how the thread will execute.

I can tell one example suppose if you want to create 2 thread one for sorting program, another for searching program. So, in the run method you have to implement the sorting algorithm may be say quick sort and another run method which basically thread for searching we can implement the binary search algorithm it is a example like this. So, in the current example we have created 3 threads to print the 3 different types of number one is negative number, even number and odd number. So, these are the threads creation.

Now, let us see how once the creates threads are created how can utilize in our program. So, this is basically the main method as we see the main method it is there name under this class these are main class. Now here we just look this things carefully what we can do is we create one thread object a of the thread; thread A. Similarly B and C are the thread objects for the 2 different thread B and thread C namely. So, here we can see 3 different threads objects are created.

Now, so, a dot start and b these are the basically once the thread object is created we can start execution of this thread. So, here basically to do this thing we have to call the start method for a b and c. Now at this point what happened is that this is the single thread that is creating this and then immediately a dot start the on thread for a thread; thread A for b thread B and c for thread C will be executed. This will print the negative number, this will print the even number and this will print the odd numbers and then finally, whenever it will come at the end of this one it will come to this system dot out print l n and it will print that that multithreading is over. So, this way the thread will execute in our program.

So, what we have learned is that there are 2 main methods, that is basically more crucial whenever the thread is there the run by the run method you have to declare that what is the task that to be accomplished by a thread and then this start method. So, start method is already defined here there in the java dot lang dot class. So, just we have to use it to start the execution of the thread. So, this is the concept about the thread creation and then finally, execution of thread ok. So, we have learned about how the thread can be created using thread class and in our next module we will discussed about other way of running threads and managing the threads.

Thank you very much.