

# Java Multithreading Part1

Syllabus

=====

1. Introduction.
2. The ways to define, instantiate and start a new Thread.
  1. By extending Thread class
  2. By implementing Runnable interface
  3. Thread class constructors
  4. Thread priority
5. Getting and setting name of a Thread.
6. The methods to prevent(stop) Thread execution.
  1. yield()
  2. join()
  3. sleep()
7. Synchronization.
8. Inter Thread communication.
9. Deadlock
10. Daemon Threads.
11. Various Conclusion
  1. To stop a Thread
  2. Suspend & resume of a thread
3. Thread group

—

Nihalnijainpath

File Edit View

22°C Light rain

Q Search

100% Windows (CPU) UTF-8

19:38 01-12-2022

File Edit View

100

5. Getting and setting name of a Thread.
  6. The methods to prevent(stop) Thread execution.
    1. yield()
    2. join()
    3. sleep()
  7. Synchronization.
  8. Inter Thread communication.
  9. Deadlock
  10. Daemon Threads.
  11. Various Conclusion
    1. To stop a Thread
    2. Suspend & resume of a thread
    3. Thread group
    4. Green Thread
    5. Thread Local
  12. Life cycle of a Thread

Light rain  
22°C

A vertical stack of Windows taskbar icons, including Start, Task View, File Explorer, Mail, Photos, Videos, and others.

110% Windows (C:\HFS) UTF-8

1.12.2022 Threads-Classmate\_Notepad

File Edit View

NitinPatilpanth



Windows (CEP) 100% Windows (CEP) 100%  
UTF-8  
ENG ENG  
IN IN  
0:12:2022 0:12:2022

1.12.2022 Threads-Classmate\_Notepad

File Edit View

NitinPatilpanth



Windows (CEP) 100% Windows (CEP) 100%  
UTF-8  
ENG ENG  
IN IN  
0:12:2022 0:12:2022

1.12.2022 Threads-Classmate\_Notepad

File Edit View

NitinPatilpanth



Windows (CEP) 100% Windows (CEP) 100%  
UTF-8  
ENG ENG  
IN IN  
0:12:2022 0:12:2022

1.12.2022 Threads-Classmate\_Notepad

File Edit View

NitinPatilpanth



Windows (CEP) 100% Windows (CEP) 100%  
UTF-8  
ENG ENG  
IN IN  
0:12:2022 0:12:2022

1.12.2022 Threads-Classmate\_Notepad

File Edit View

NitinPatilpanth



Windows (CEP) 100% Windows (CEP) 100%  
UTF-8  
ENG ENG  
IN IN  
0:12:2022 0:12:2022

### Process based multitasking

=====

Executing several task simultaneously is the concept of multitasking.

There are 2 types of Multitasking.

- a. Process based multitasking
- b. Thread based multitasking.

eg:: typing a java pgm  
listening to a song  
downloading the file from internet

Process based multitasking is best suited at "os level".

Executing several tasks simultaneously where each task is a separate independent process such type of multitasking is called "process based multitasking".

Ln 42 Col 57 29C Light Rain Q. Search 100% Windows (CEP) 100% Windows (CEP) 100%  
UTF-8  
ENG ENG  
IN IN  
0:12:2022 0:12:2022

1.12.2022 Threads ClassNotes - Notepad

File Edit View

**Process based multitasking**

=====

Executing several tasks simultaneously where each task is a separate independent process such type of multitasking is called "process based multitasking".

eg:: typing a java pgm  
listening to a song  
downloading the file from internet

Process based multitasking is best suited at "os level".

**Thread based multitasking**

=====

=> Executing several tasks simultaneously where each task is a separate independent part of the same Program, is called "Thread based MultiTasking".



112 2022

Light Rain

20°C

Q. Search

10% Windows (CEP) UTF-8

ENG IN 01-12-2022

1.12.2022 Threads-Classmate\_Notepad

File Edit View

listening to a song  
downloading the file from internet

NiharikaJain

Process based multitasking is best suited at "os level".

Thread based multitasking

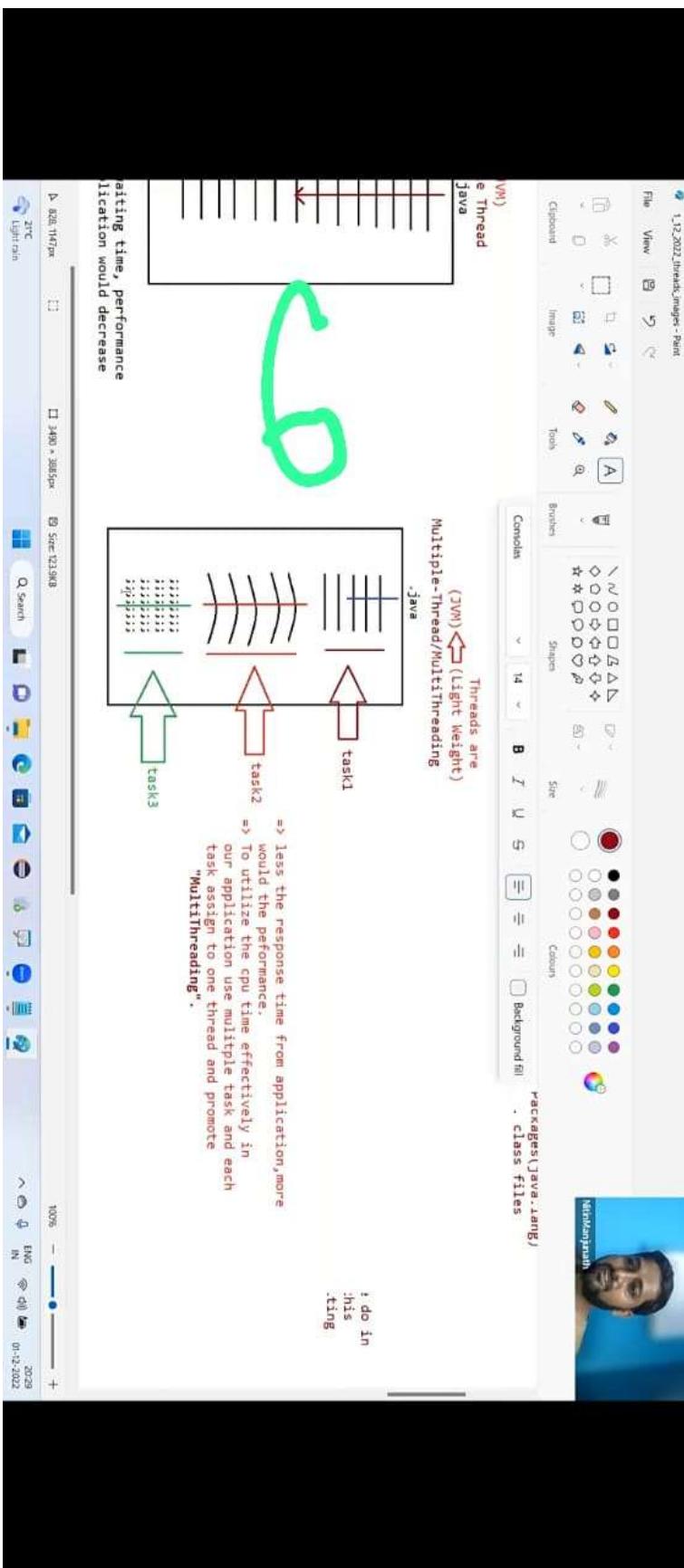
=====  
=> Executing several tasks simultaneously where each task is a separate independent part of the same Program, is called "Thread based MultiTasking".  
Each independent part is called "Thread".

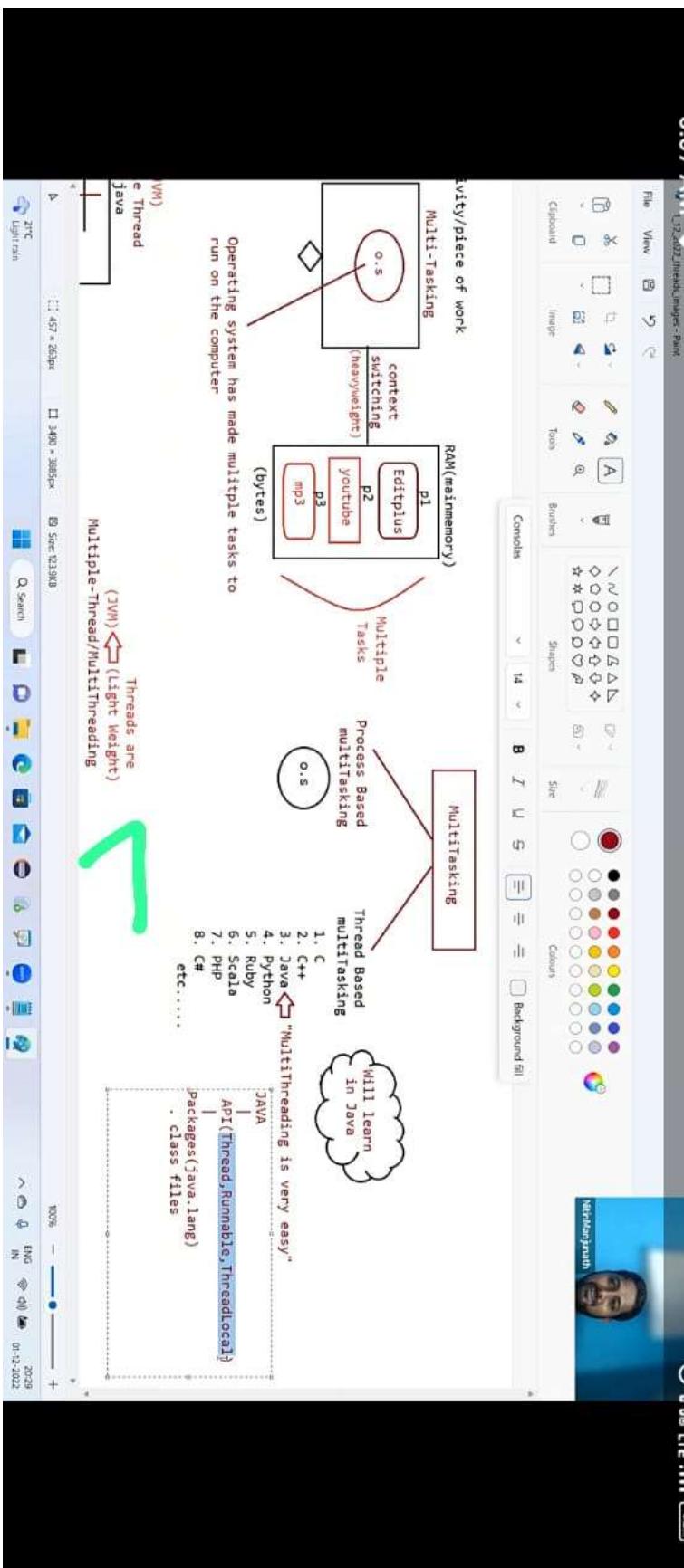
1. This type of multitasking is best suited at "Programmatic level".  
The main advantages of multitasking is to reduce the response time of the system and to improve the performance.

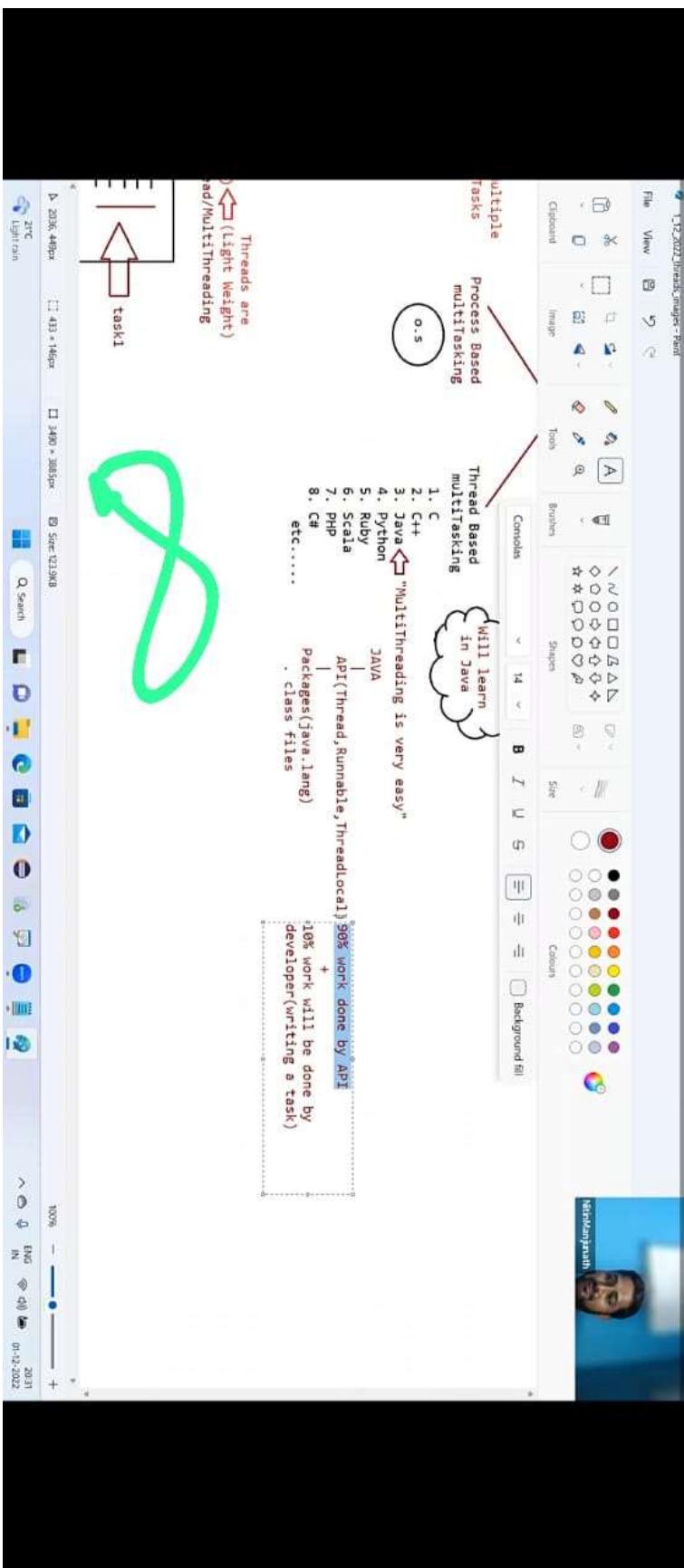
2. The main important application areas of multithreading are

- a. To implement multimedia graphics
- b. To develop web application servers
- c. To develop video games









6:10 AM 1.0, 502 Threads, 2 Cores, Notepad

File Edit View

**Thread based multitasking**

=====

=> Executing several tasks simultaneously where each task is a separate independent part of the same Program, is called "Thread based Multitasking".

Each independent part is called "Thread".

1. This type of multitasking is best suited at "Programmatic level".  
The main advantages of multitasking is to reduce the response time of the system and to improve the performance.

2. The main important application areas of multithreading are

- To implement multimedia graphics
- To develop web application servers(will learn in JEE)
- To develop video games<sup>1</sup>
- To develop animations

3. Java provides inbuilt support to work with threads through API called Thread, Runnable, ThreadGroup, ThreadLocal,...

4. To work with multithreading, java developers will code only for 10% remaining 90% java API will take care..



⑤ 6:10 AM LTE 0.11 1.57

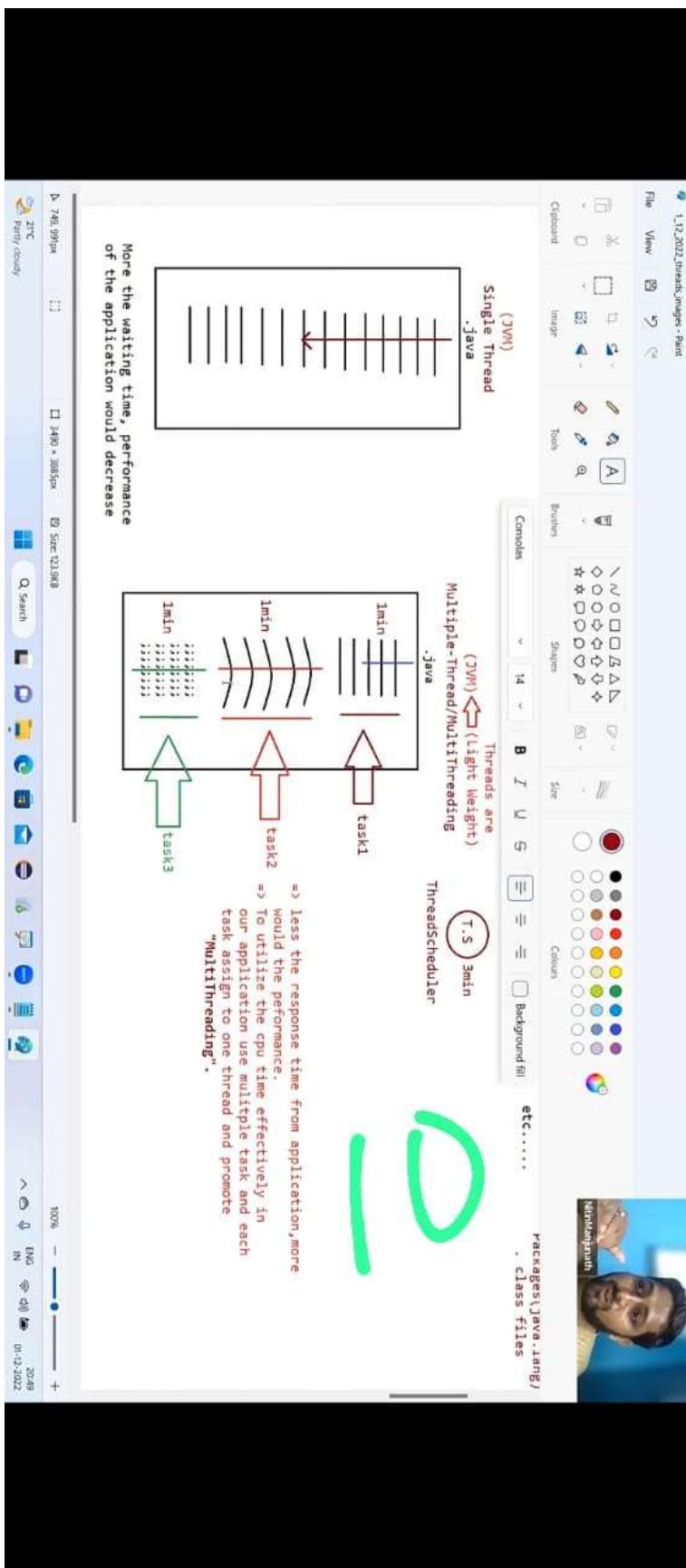
Ln 57 Col 30

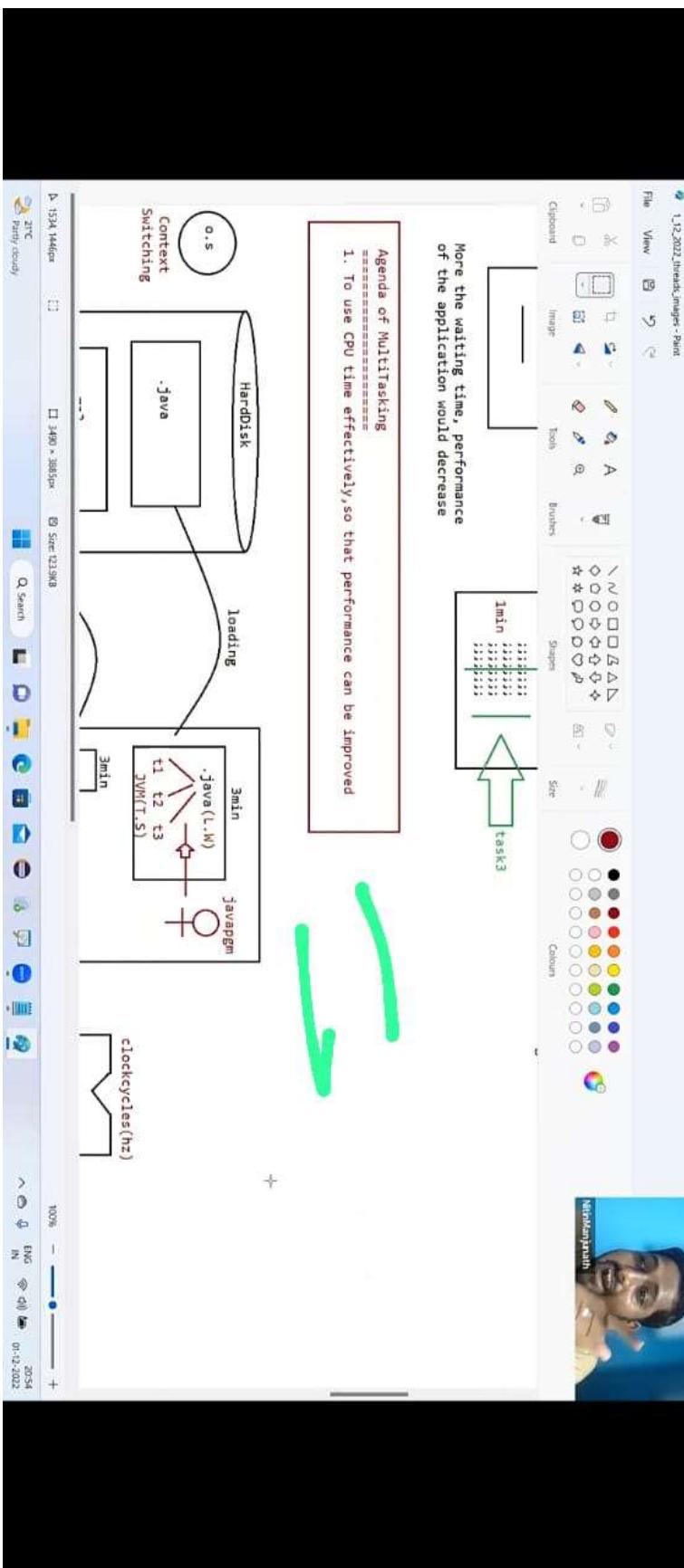
20°C Light Rain

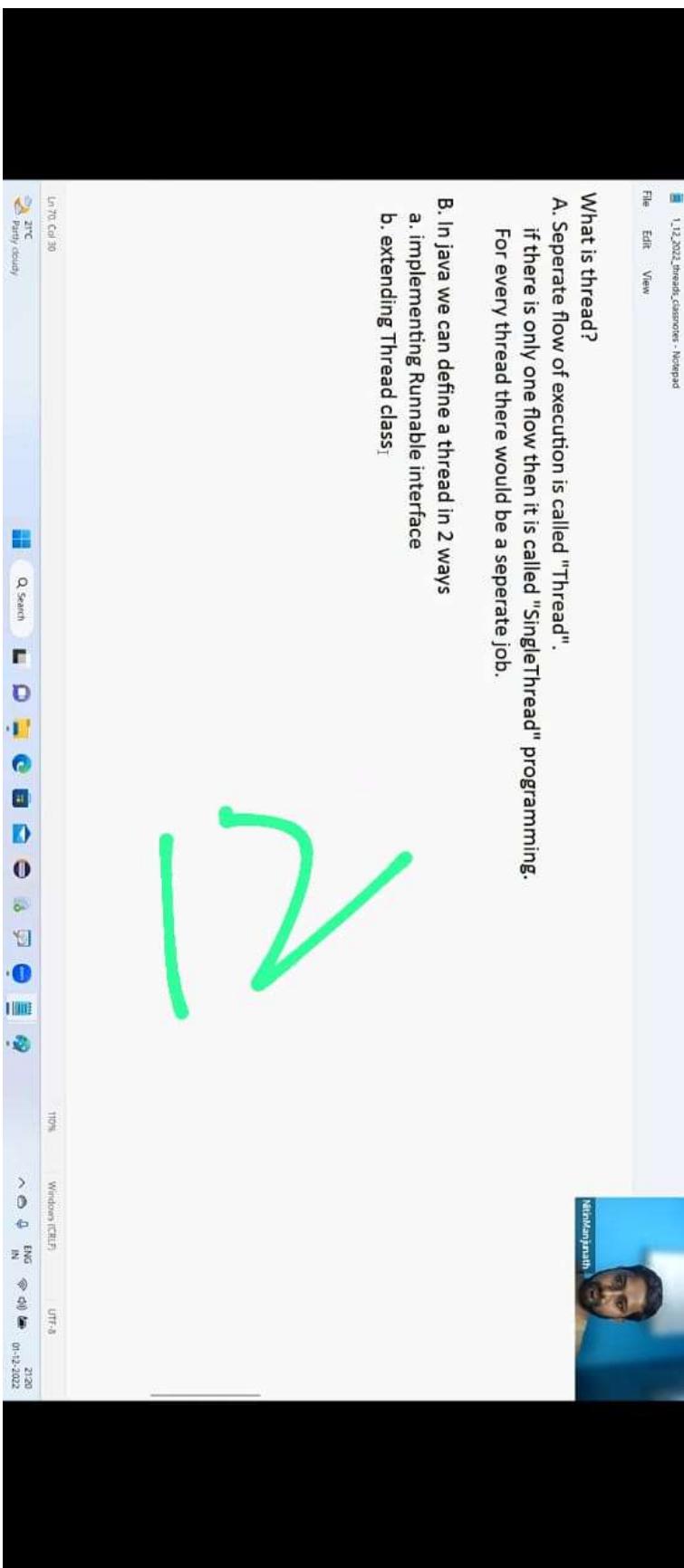
Q. Search

100% Windows (CEP) | UTF-8

^ ⌂ ⌂ ENG ⌂ d) 🔍 0:12:2033  
IN







Editor - (D:\Whisper classes\TextJava1)

File Edit View Search Document Project Icons Browser Ethernet Window Help

Directory Object Folders

[D] New Volume D:\ Whisper classes

10      1 System.out.println("Child thread");

11      2

12      3

13      4

14      5

15 public class Test {

16     public static void main(String[] args){

17         MyThread t = new MyThread();

18         t.start();

19     }

20     // 2 threads started and eagerly waiting for CPU time, scheduling is done by T.S

21     //task for main thread

22     for (int i = 1;i<=10;i++)

23     {

24         System.out.println("Main thread");

25     }

26 }

27 }

28 }

29 }

30 }

Java (\*.java)

TestJava

For Help press F1

20°C Party cloudy

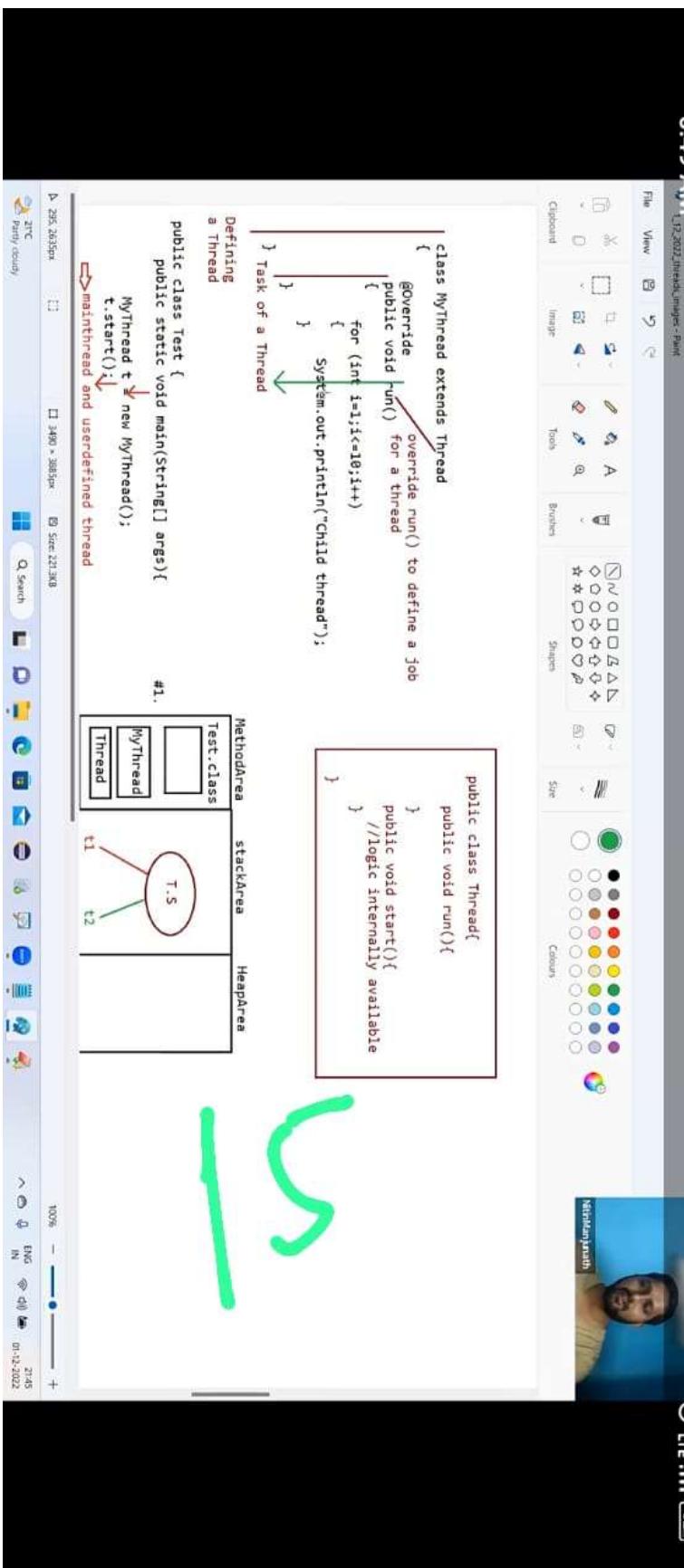
Q Search

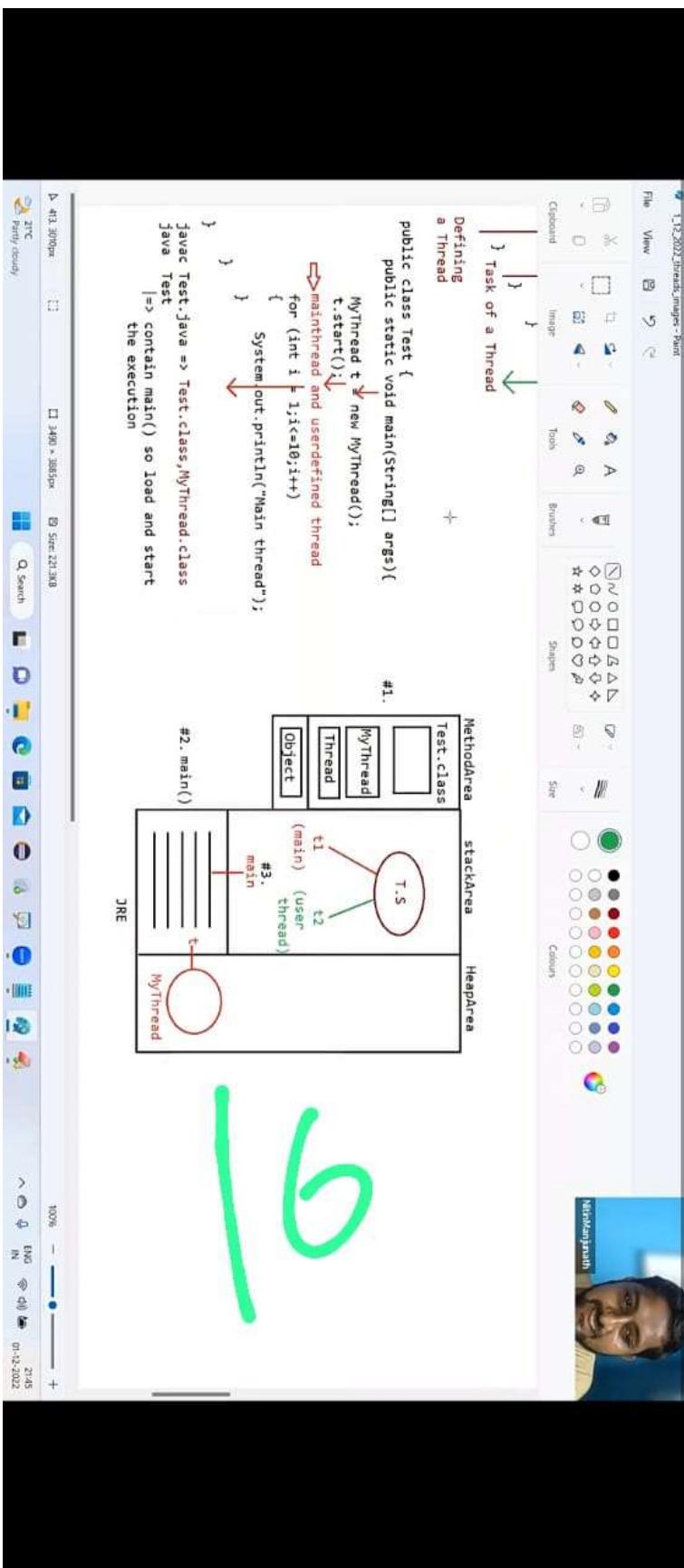
In 25 00:47 35 00 PC ANSI

^ < > ^ & EN IN

ENG 2022-01-12 2022







1.12.2022.Breaks\_classnames\_Notepad  
File Edit View  
**a. Implementing Runnable interface**  
**b. extending Thread class**

1. Extending Thread class  
=> we can create a Thread by extending a Thread.

```
class MyThread extends Thread{  
    @Override  
    public void run(){  
        for(int i=0;i<10;i++)  
            System.out.println("child thread");  
    }  
}  
defining a thread(writing a class and extending a Thread)  
job a thread(code written inside run())  
t.start(); //starting a thread
```



21°C  
Partly cloudy

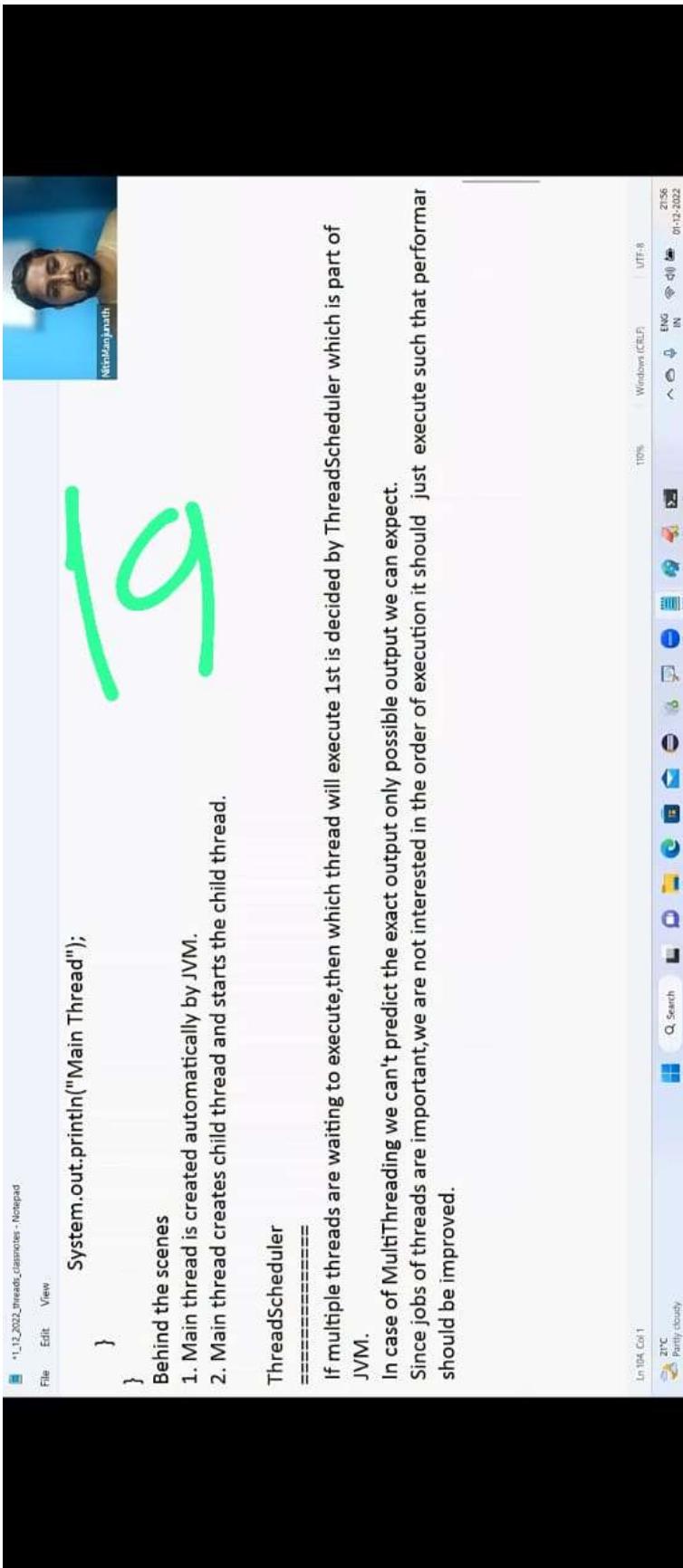


Ln 76 Col 11 100% Windows (CEP) UTF-8

ENG IN 01-12-2022

```
1.12.2022 Threads_classnotes_Nopaid  
File Edit View  
} defining a thread(writing a class and extending a Thread)  
job a thread(code written inside run())  
  
class ThreadDemo{  
    public static void main(String... args){  
        MyThread t =new MyThread(); //Thread instantiation  
        t.start(); //starting a thread  
        ;;; // At this line 2 threads are there  
  
        for(int i=1;i<=5;i++)  
            System.out.println("Main Thread");  
    }  
}  
Behind the scenes  
1. Main thread is created automatically by JVM.  
2. Main thread creates child thread and starts the child thread.
```

2



19

```
File Edit View System.out.println("Main Thread"); } Behind the scenes 1. Main thread is created automatically by JVM. 2. Main thread creates child thread and starts the child thread. ThreadScheduler ===== If multiple threads are waiting to execute, then which thread will execute 1st is decided by ThreadScheduler which is part of JVM. In case of MultiThreading we can't predict the exact output only possible output we can expect. Since jobs of threads are important, we are not interested in the order of execution it should just execute such that performance should be improved.
```



Editor - D:\Whisper classes\TutJava

File Edit View Search Document Project Tools Browser Ethernet Window Help

Directory Object Functions

[D:] New Volume D:\ Whisper classes

1 2 3 4 5 6 7

```
/*
 * 
 */
class Thread
{
    //Heart of MultiThreading
    public void start(){
        1. Register the thread with ThreadScheduler
        2. All other mandatory low level activities(memory level)
        3. invoke or call run() method
    }
    public void run()
    {
        //no implementation
    }
}
class MyThread extends Thread
@Override
public void run()
```

Java (\*.java)

Java (\*.java)

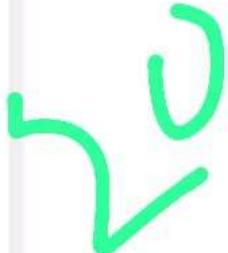
For Help press F1

20C Party cloudy

Q Search

In 9 col 70 54 00 PC ANSI 100% (100%)

^ < > & EN IN 01-12-2022 22:05



EdBans - D:\Whisper class\T.java

File Edit View Search Document Project Icons Brower Emitter Window Help

Directory Object Functions

[D:] New Volume D:\ Whispers classes

Test.java

```
1 public class Test {  
2     public static void main(String[] args){  
3         MyThread t = new MyThread();  
4         //This line will create a new Thread which is responsible to execute run().  
5         t.start();  
6     }  
7     // 2 threads started and eagerly waiting for CPU time, scheduling is done by T.S  
8     //task for main thread  
9     for (int i = 1;i<=10;i++)  
10    {  
11        System.out.println("Main thread");  
12    }  
13}
```

Java (J2SE) TestJava For Help press F1 20°C Party cloudy

In 9 col 70 54 00 PC ANSI 100% 0:12:22 22:05 ENG ⇧ d) 0:12:22

1\_12\_2022\_threads\_classnotes - Notepad

case2: diff b/w t.start() and t.run()

If we call `t.start()` and separate thread will be created which is responsible to execute `run()` method.

If we call `t.run()`, no separate thread will be created rather the method will be called just like normal method by main thread. If we replace `t.start()` with `t.run()` then the output of the program would be

```
child thread  
main thread  
main thread  
main thread
```

22

1.12.2022 Threads\_classnotes\_NotePad

File Edit View

**case3:: Importance of Thread class start() method**

For every thread, required mandatory activities like registering the thread with ThreadScheduler will be taken by Thread class start() method and programmer is responsible of just doing the job of the Thread inside run() method.

start() acts like an assistance to programmer.

```
public void start()
{
    register thread with ThreadScheduler
    All other mandatory low level activities
    invoke or calling run() method.
}
```

We can conclude that without executing Thread class start() method there is no chance of starting a new Thread in java.

Due to this start() is considered as heart of MultiThreading.



We can conclude that without executing Thread class start() method there is no chance of starting a new Thread. Due to this start() is considered as heart of MultiThreading.

case4:: If we are not overriding run() method if we are not Overriding run() method then Thread class run() method will be executed which has empty implementation and hence we wont get any output.

eg::

```
class MyThread extends Thread{  
    public static void main(String... args){  
        MyThread t=new MyThread();  
        t.start();  
    }  
}
```

It is highly recommended to override run() method, otherwise don't go for MultiThreading concept.

1.12.2022 import person classes - Notepad

File Edit View

```
class Payload {
    private int weight;
    public Payload (int w) { weight = w; }
    public void setWeight(int w) { weight = w; }
    public String toString() { return Integer.toString(weight); }
}

public class TestPayload {
    static void changePayload(Payload p) {
        /* Insert code */ //Line 12
    }

    public static void main(String[] args) {
        Payload p = new Payload(200); // weight = 200
        p.setWeight(1024); // weight = 1024
        changePayload(p);
        System.out.println("p is " + p);
    }
}
```

Which code fragment, inserted at the end of line 12, produces the output `p is 420?`

A. `p.setWeight(420);`  
B. `p.changePayload(420);`  
C. `p = new Payload(420);`  
D. `Payload.setWeight(420);`

20C Party country

Q. Search

100% Windows (CST) UTF-8

^ ⌂ ⌄ ENG ⌂ ⌄ 0:12:22:26 0:12:2022

```
1.12.2022.Anupret.Session.Classes - Notepad  
File Edit View  
  
public class TestPayload {  
    static void changePayload(Payload p){  
        p.setWeight(420);  
        /* insert code */ //Line 12  
    }  
  
    public static void main(String[] args){  
        Payload p = new Payload(200);// weight = 200  
        p.setWeight(1024);// weight = 1024  
        changePayload(p);  
        System.out.println("p is " + p);  
    }  
}
```

Which code fragment, inserted at the end of line 12, produces the output p is 420?

- A. p.setWeight(420);
- B. p.changePayload(420);
- C. p = new Payload(420);
- D. Payload.setWeight(420);
- E. p.Payload.setWeight(420);

Answer: A

2b

20C  
Party cloudy



Ln 26 Col 1  
100% Windows (UTF-8) UTF-8  
^ ⌂ ↕ ENG IN 22:27 01-12-2022



1.12.2022-answers-session\_classes - Notepad

File Edit View

Question

```
abstract class C1 {  
    public C1() { System.out.print(1); }  
}  
class C2 extends C1 {  
    public C2() { System.out.print(2); }  
}  
class C3 extends C2 {  
    public C3() { System.out.println(3); }  
}  
public class Ctest {  
    public static void main(String[] a) { new C3(); }  
}
```

What is the result?

- A. 3
- B. 23
- C. 32
- D. 123
- E. 321
- F. Compilation fails.
- G. An exception is thrown at runtime.

Q. Search          

Ln 54 Col 34  
20C Party cloudy

100%

Windows (CET)

UTF-8

EN

IN

22:33

01-12-2022





```
1.12.2022-MyPrestoSession-classnames - Notepad
File Edit View
}
class C2 extends C1{
}
public C2(){ System.out.print(2); }
}
class C3 extends C2{
}
public C3(){ System.out.println(3); }
}
public class Ctest{
}
public static void main(String[] a){ new C3(); }

What is the result?
A. 3
B. 23
C. 32
D. 123
E. 321
F. Compilation fails.
G. An exception is thrown at runtime.
```

Answer: D

```
1.12.2022.properties-and-classes - Notepad  
File Edit View  
Given:  
class One{  
    public One foo(){  
        return this;  
    }  
}  
class Two extends One{  
    public One foo(){  
        return this;  
    }  
}  
class Three extends Two{  
    // Insert method here  
}
```

Which two methods, inserted individually, correctly complete the Three class? (Choose two.)

- A. public void foo(){}  
B. public int foo(){return 3;}  
C. public Two foo(){return this;}  
D. public One foo(){return this;}  
E. public Object foo(){return this;}

PP

20°C  
Partly cloudy



Ln 76 Col 4  
100% Windows (CET)  
UTF-8  
EN IN 01-12-2022 22:38

```
1.12.2022 snippets-session_classes - Notepad  
File Edit View  
return this;  
}  
}  
class Two extends One {  
    public One foo() {  
        return this;  
    }  
}  
class Three extends Two {  
    // Insert method here  
}
```

Which two methods, inserted individually, correctly complete the Three class? (Choose two.)

- A. public void foo() {}
- B. public int foo() {return 3;}
- C. public Two foo() {return this;}
- D. public One foo() {return this;}
- E. public Object foo() {return this;}

Answer : D



20°C  
Partly cloudy

Q. Search

Ln 99 Col 1

100% Windows (C:\LT)

UTF-8

ENGLISH

IN

22:39

01-12-2022

```
File Edit View  
1,1,0,0,22,unsorted,lesson01,charmers - Notepad  
  
class Two extends One {  
    public One foo()  
    {  
        return this;  
    }  
}  
  
class Three extends Two{  
    // insert method here  
}  
  
Which two methods, inserted individually, would result in a compilation error?  
A. public void foo() {}  
B. public int foo() { return 3; }  
C. public Two foo() { return this; }  
D. public One foo() { return this; }  
E. public Object foo() { return this; }
```

Which two methods, inserted individually, correctly complete the Three class? (Choose two.)

- ```
B. public int foo() {return 3;}  
C. public Two foo() {return this;}  
D. public One foo() {return this;}
```

**Answer :** C,D

IS-A relationship  
Child

## Covariant return types



In 76 Col 12

## Covariant return types

IS-A relationship  
Child

**Answer :** C,D

Which two methods, inserted individually, correctly complete the Three class? (Choose two.)

- ```
B. public int foo() {return 3;}  
C. public Two foo() {return this;}  
D. public One foo() {return this;}
```

In 76 Col 12

## Covariant return types

IS-A relationship  
Child

**Answer :** C,D

Which two methods, inserted individually, correctly complete the Three class? (Choose two.)

- ```
B. public int foo() {return 3;}  
C. public Two foo() {return this;}  
D. public One foo() {return this;}
```

1.12.2022 snippet\_selections - Notepad

File Edit View

Q>

Given:

```
public interface A { public void m1(); }

class B implements A {} //CE

class C implements A { public void m1() {} }

class D implements A { public void m1(int x) {} } //CE

abstract class E implements A {}

abstract class F implements A { public void m1() {} }

abstract class G implements A { public void m1(int x) {} }
```

What is the result?

A. Compilation succeeds.  
B. Exactly one class does NOT compile.  
C. Exactly two classes do NOT compile.  
D. Exactly four classes do NOT compile.

Ln 128 Col 9

21°C Party cloudy

Q. Search

100% Windows (CET) UTF-8

EN IN 01-12-2022 22:46

```
1.12.2022 import java.util.*; - Notepad  
File Edit View  
class D implements A { public void m1(int x) {} } //CE  
abstract class E implements A {}  
abstract class F implements A { public void m10() {} }  
abstract class G implements A { public void m1(int x) {} }  
  
What is the result?  
A. Compilation succeeds.  
B. Exactly one class does NOT compile.  
C. Exactly two classes do NOT compile.  
D. Exactly four classes do NOT compile.  
E. Exactly three classes do NOT compile
```

Answer: C

X

20°C  
Partly cloudy



11:19, Oct 1

100% Windows (C:\EF7)

UTF-8

ENG IN 01-12-2022 22:47

192.168.1.102~angies-session-classnames - Notepad

File Edit View

**Answer: C**

Q>

Given:

```
1. class TestA {  
2.     public void start() { System.out.println("TestA"); }  
3. }  
4. public class TestB extends TestA {  
5.     public void start() { System.out.println("TestB"); } //overridden method  
6.     public static void main(String[] args) {  
7.         ((TestA)new TestB()).start();  
8.     }  
9. }
```

What is the result?

- A. TestA
- B. TestB
- C. Compilation fails.
- D. An exception is thrown at runtime.

Answer: B



192.168.1.102~angies-session-classnames - Notepad

File Edit View

**Answer: C**

Q>

Given:

```
1. class TestA {  
2.     public void start() { System.out.println("TestA"); }  
3. }  
4. public class TestB extends TestA {  
5.     public void start() { System.out.println("TestB"); } //overridden method  
6.     public static void main(String[] args) {  
7.         ((TestA)new TestB()).start();  
8.     }  
9. }
```

What is the result?

- A. TestA
- B. TestB
- C. Compilation fails.
- D. An exception is thrown at runtime.

Answer: B



```
1.12.2022.Anupam.Session.Classes - Notepad  
File Edit View  
Given:  
class Line{  
    public class Point{  
        public int x, y;  
    }  
    public Point getPoint(){  
        return new Point();  
    }  
}  
class Triangle{  
    public Triangle(){  
        // insert code here line 16  
        Line.Point p = new Line().getPoint();  
    }  
}
```

Which code, inserted at line 16, correctly retrieves a local instance of a Point object?

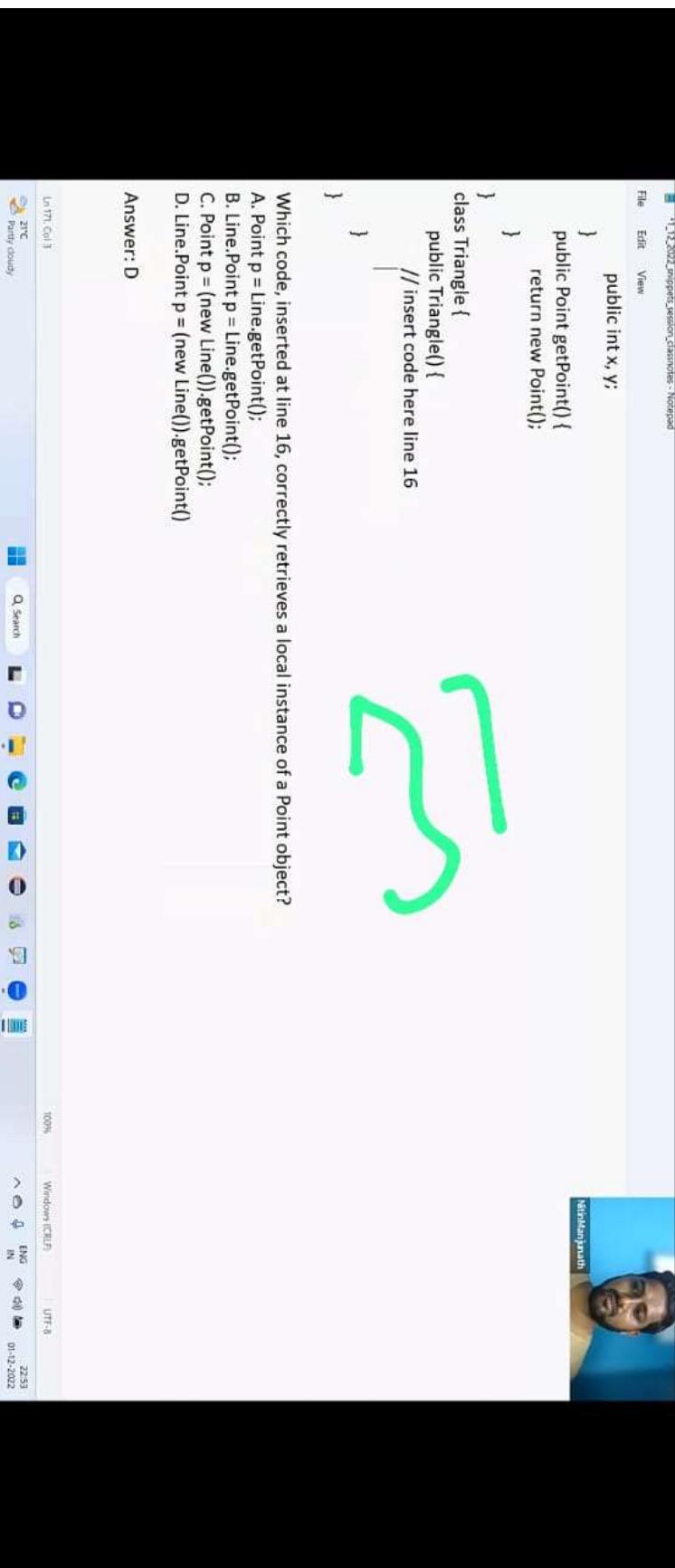
- A. Point p = Line.getPoint();
- B. Line.Point p = Line.getPoint();
- C. Point p = (new Line()).getPoint();
- D. Line.Point p = (new Line()).getPoint()

36

21°C  
Partly cloudy



Ln 150, Col 10      100%      Windows (CET)      UTF-8  
^ ⌂ ⌄ ENG ⌄ d) ↵ 0:12:22:53  
In



```
1 12/2022/progression_of_descriptions - Notepad
File Edit View
public int x, y;
}
public Point getPoint() {
    return new Point();
}
class Triangle {
    public Triangle() {
        // insert code here line 16
    }
}
```

Which code, inserted at line 16, correctly retrieves a local instance of a Point object?

- A. Point p = Line.getPoint();
- B. Line.Point p = Line.getPoint();
- C. Point p = (new Line()).getPoint();
- D. Line.Point p = (new Line()).getPoint()

Answer: D