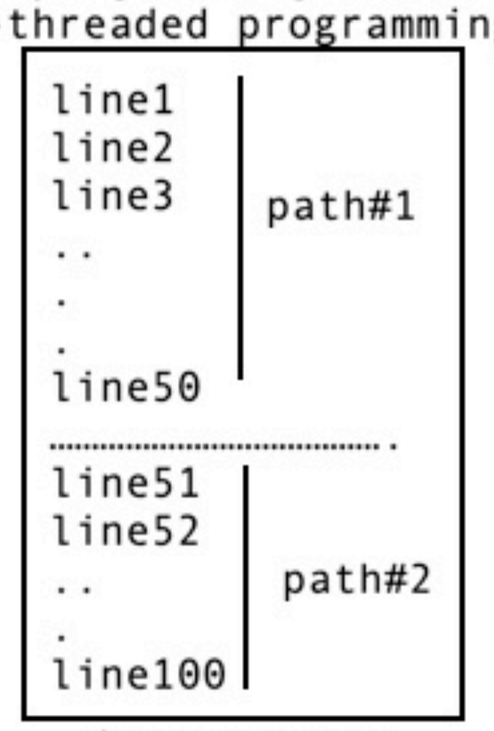
programming model:
multi-threaded programming model

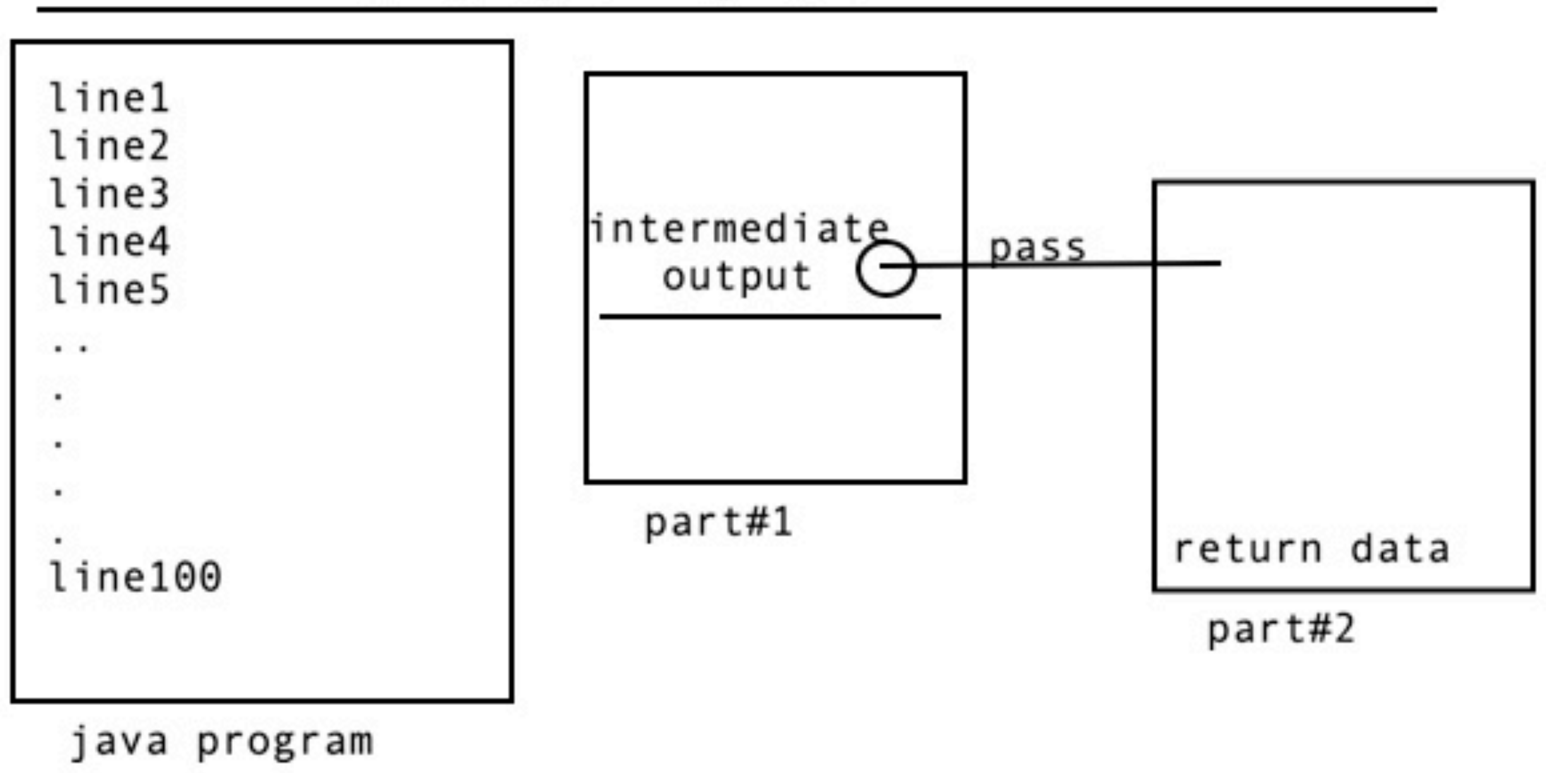


java program

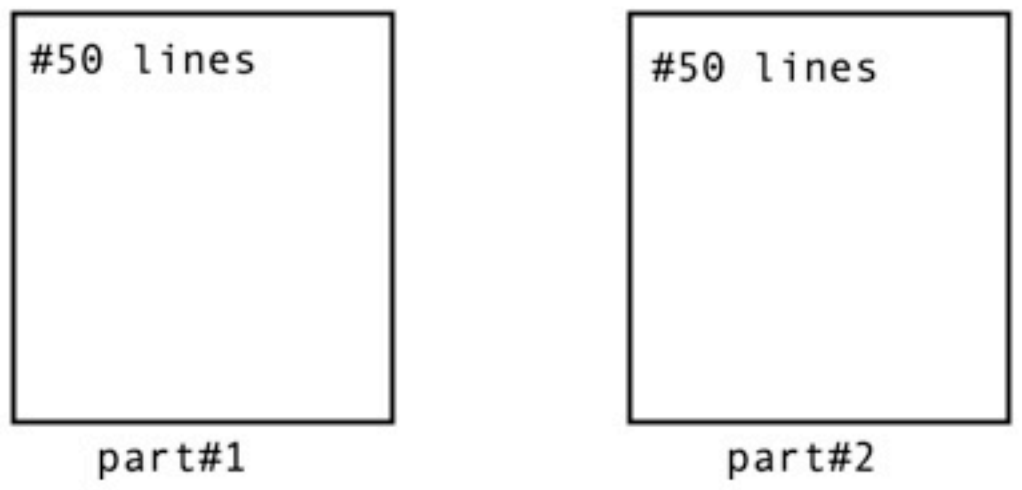
1. api/interface driven programming model
1.1 linear programming (synchronous/blocking programming)
1.2 multi-threaded programming

2. event-driven programming model
2.1 asynchronous programming (preferrably)
2.2 synchronous programming

programming model:
linear programming model



in the above program we have #2 independent paths of execution written in one single program or class or a component, but jvm executes the above code linearly only.



instead we can break down the program into #2 separate components, indicating #2 paths or parts of the code.
even though we broken the code into #2 separate components, still those will be executed linearly only, because the existing programming constructs that are built into the language are designed to work as linear programming only.
For eg.. when one method(callee) invokes the another method (caller), the control will be transferred to the caller blocking the execution of the callee until the caller completes execution and returns an return value, the callee will not be resumed that is how the programs are designed to be executed.

But we wanted these 2 parts/paths of the program to be executed independently without waiting one for the another. So indicate the same to the jvm, java has introduced Multi-threaded programming api.

```
class Part1 extends Thread {
  public run() {
    #50 lines of code
  }
}
```

```
class Part2 extends Thread {
  public run() {
    #50 lines of code
  }
}
```

#Main

```
invokes Part1
invokes Part2

new Part1().start();
new Part2().start();
```