6 Concurrency - 2

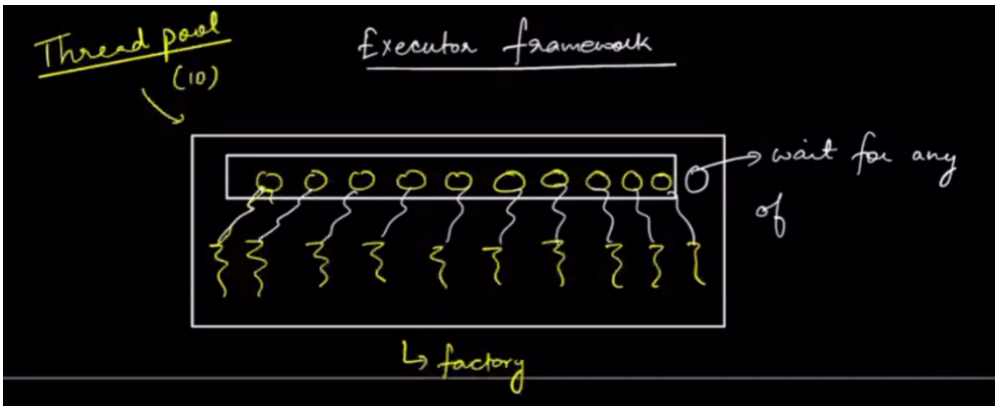
04 June 2024 21:15

Agenda :

1. Executor
2. Callable
3. Multithreaded Merge Sort
4. Intro to adder Subtractor Problem

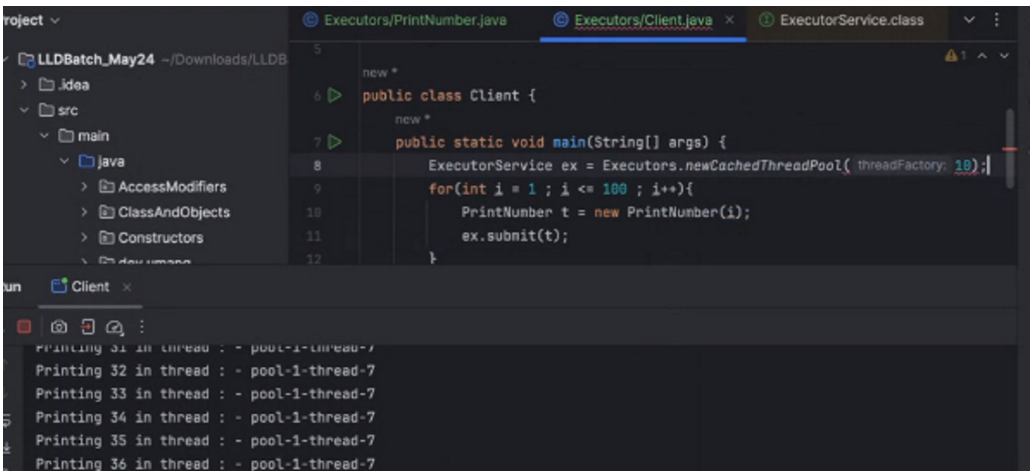
1. Executors

- In a multithreaded environment, we divide the responsibilities into Parts
  a. **Client :** knows What task to run
  b. **Executors :** knows the best way to efficiently run the task, in order to achieve concurrency.

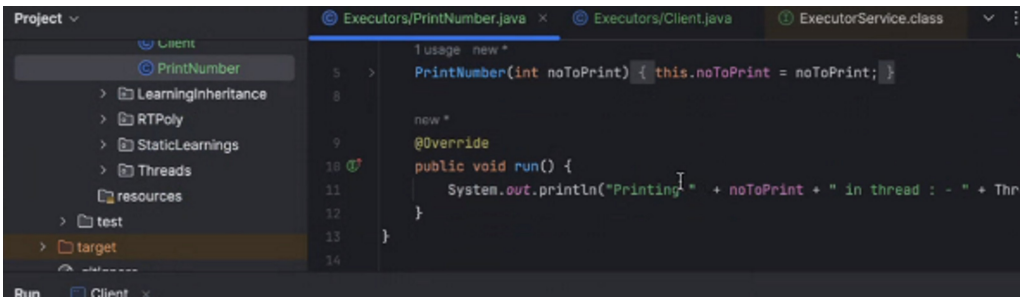


Waits for any other thread to finish to assign other task.

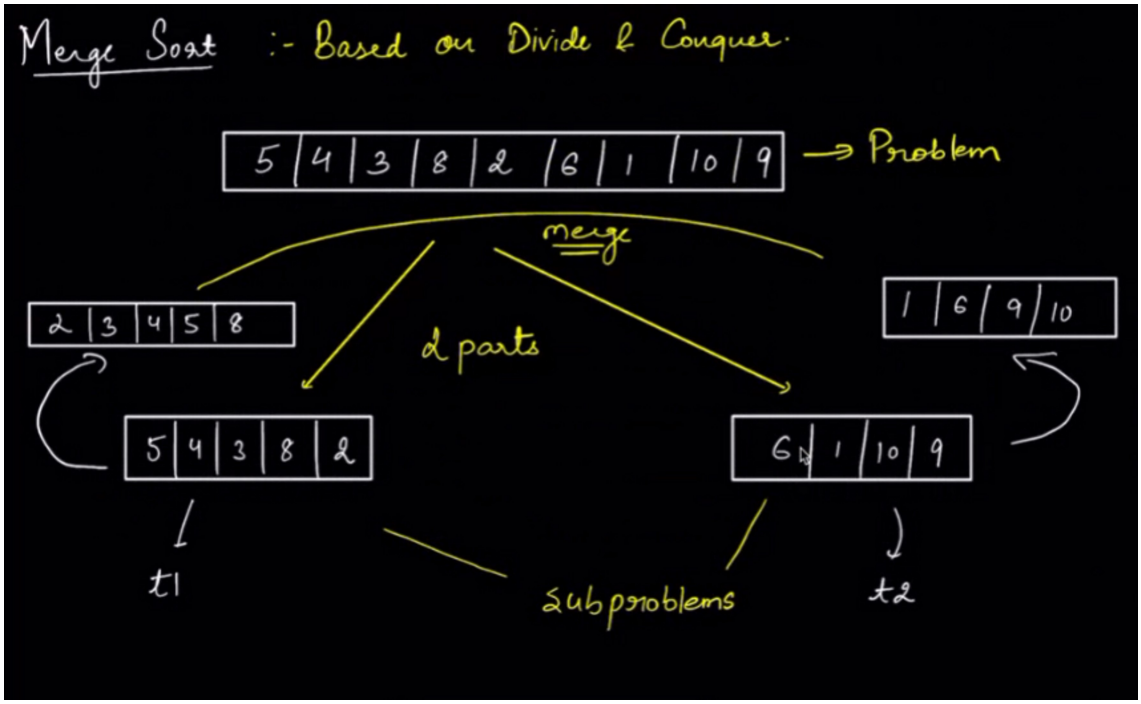
- Example of Executor Implementation



Newcatchedthread pool passed with parameter creates that many threads.



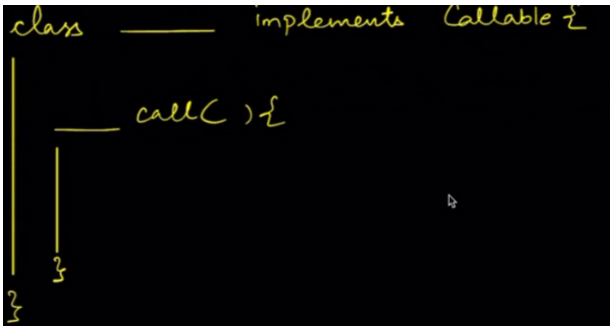
Merge Sort with Threads to perform sorting concurrently



We cannot use run(), (runnable interface) as it does not return anything.
So we need Callable interface here in this case.

2. Callable

Callable : Runnable + Returns some data.



3. Multithreaded Merge Sort

Client.java

```
package MergeSortMultiThreaded;

import java.util.List;
import java.util.concurrent.ExecutionException;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
import java.util.concurrent.Future;


public class Client {
   public static void main(String[] args) throws ExecutionException, InterruptedException {
      ExecutorService ex = Executors.newCachedThreadPool();

      List<Integer> ls = List.of(1,4,56,3,2,43,2,3,32,23,3);
```

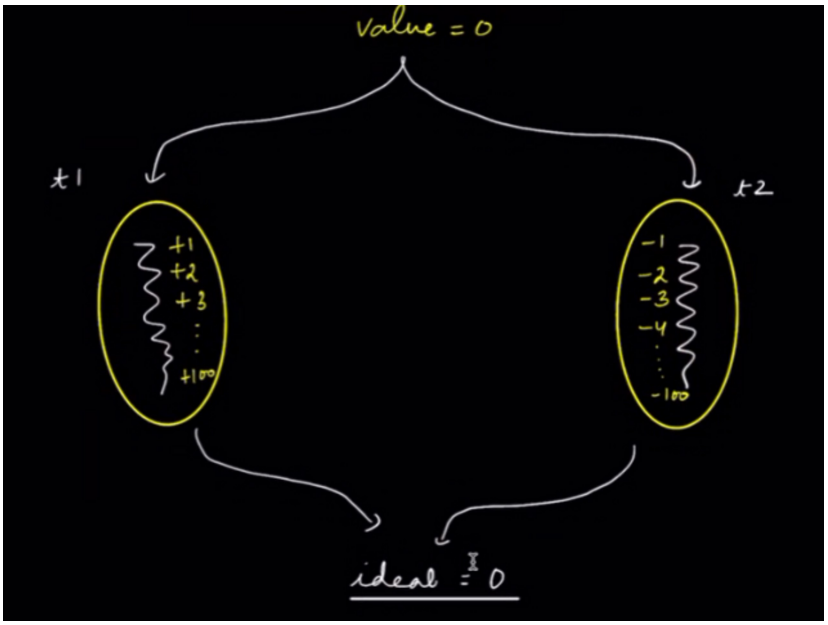
```
        Sorter t = new Sorter(ls, ex);

        Future<List<Integer>> res = ex.submit(t);

        ls = res.get();

        System.out.println(ls);

    }
}
```

Sorter.java

```
package MergeSortMultiThreaded;

import java.util.ArrayList;
import java.util.List;
import java.util.concurrent.Callable;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Future;

public class Sorter implements Callable<List<Integer>> {
    List<Integer> arrayToSort;
    ExecutorService ex;
    Sorter(List<Integer> arrayToSort, ExecutorService ex){
        this.arrayToSort = arrayToSort;
        this.ex = ex;
    }
    @Override
    public List<Integer> call() throws Exception {
        //Write the entire merge sort code
        if(arrayToSort.size() <= 1){
            return arrayToSort;
        }

        int mid = arrayToSort.size() / 2;
        /*
        First half - 0 to mid - 1
        second half - mid to size - 1
         */

        List<Integer> leftHalf = new ArrayList<>();
        for(int i = 0 ; i < mid ; i++){
            leftHalf.add(arrayToSort.get(i));
        }

        List<Integer> rightHalf = new ArrayList<>();
        for(int i = mid ; i < arrayToSort.size() ; i++){
            rightHalf.add(arrayToSort.get(i));
        }

        Sorter task1 = new Sorter(leftHalf, ex);
        Sorter task2 = new Sorter(rightHalf, ex);

        Future<List<Integer>> leftSortedArray = ex.submit(task1);
        Future<List<Integer>> rightSortedArray = ex.submit(task2);
        leftHalf = leftSortedArray.get();
        rightHalf = rightSortedArray.get();

        /*
        merge left and right half
         */
```

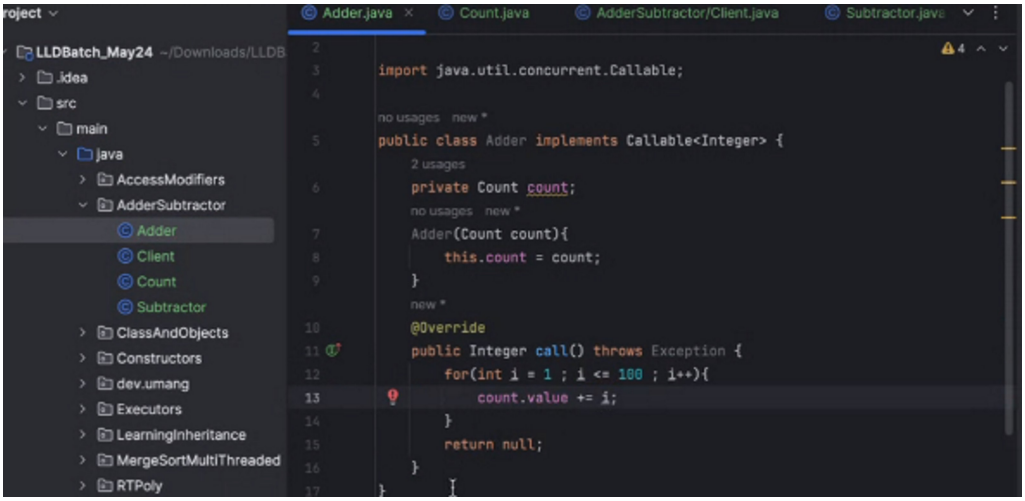
```
    List<Integer> finalMergedArray = new ArrayList<>();

    int i = 0 , j = 0;
    while(i < leftHalf.size() && j < rightHalf.size()){
      if(leftHalf.get(i) < rightHalf.get(j)){
        finalMergedArray.add(leftHalf.get(i));
        i++;
      }else{
        finalMergedArray.add(rightHalf.get(j));
        j++;
      }
    }

    while(i < leftHalf.size()){
      finalMergedArray.add(leftHalf.get(i));
      i++;
    }

    while(j < rightHalf.size()){
      finalMergedArray.add(rightHalf.get(j));
      j++;
    }

    return finalMergedArray;
  }
}
```

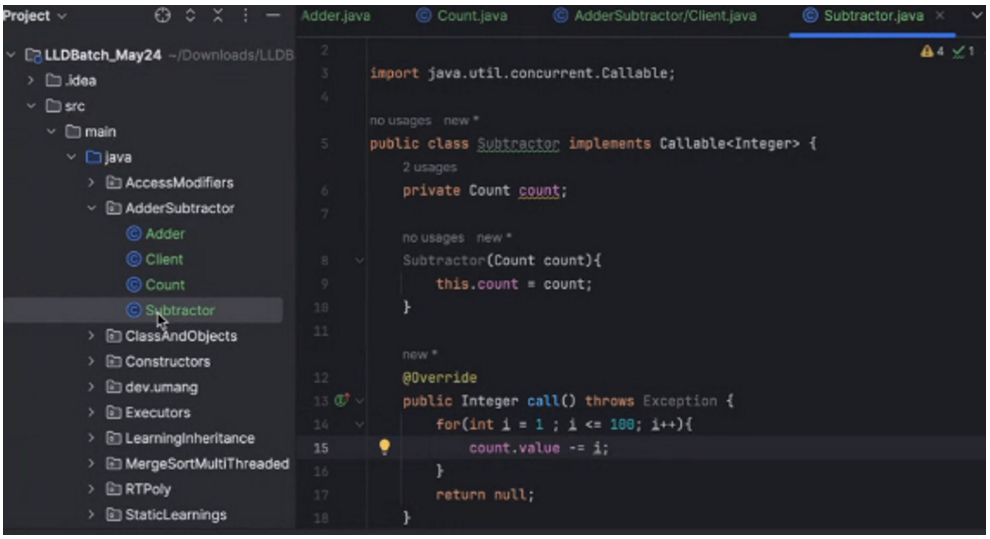
4. Synchronization Problem
   (Adder Subtractor Problem)



But since both the threads are not synchronized result need not to be 0.

Code :

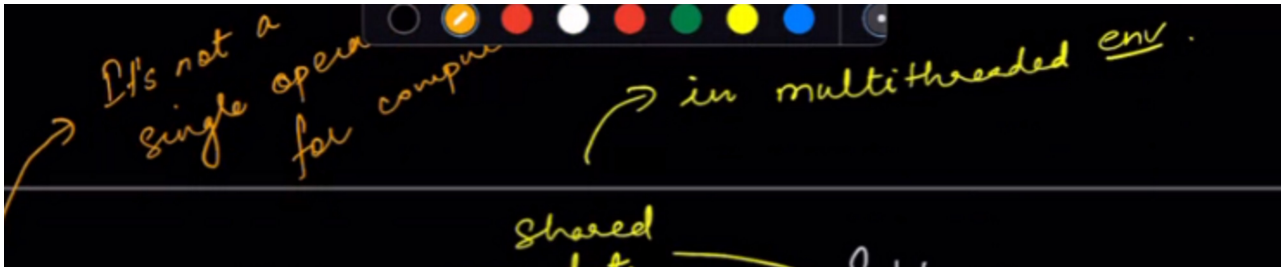


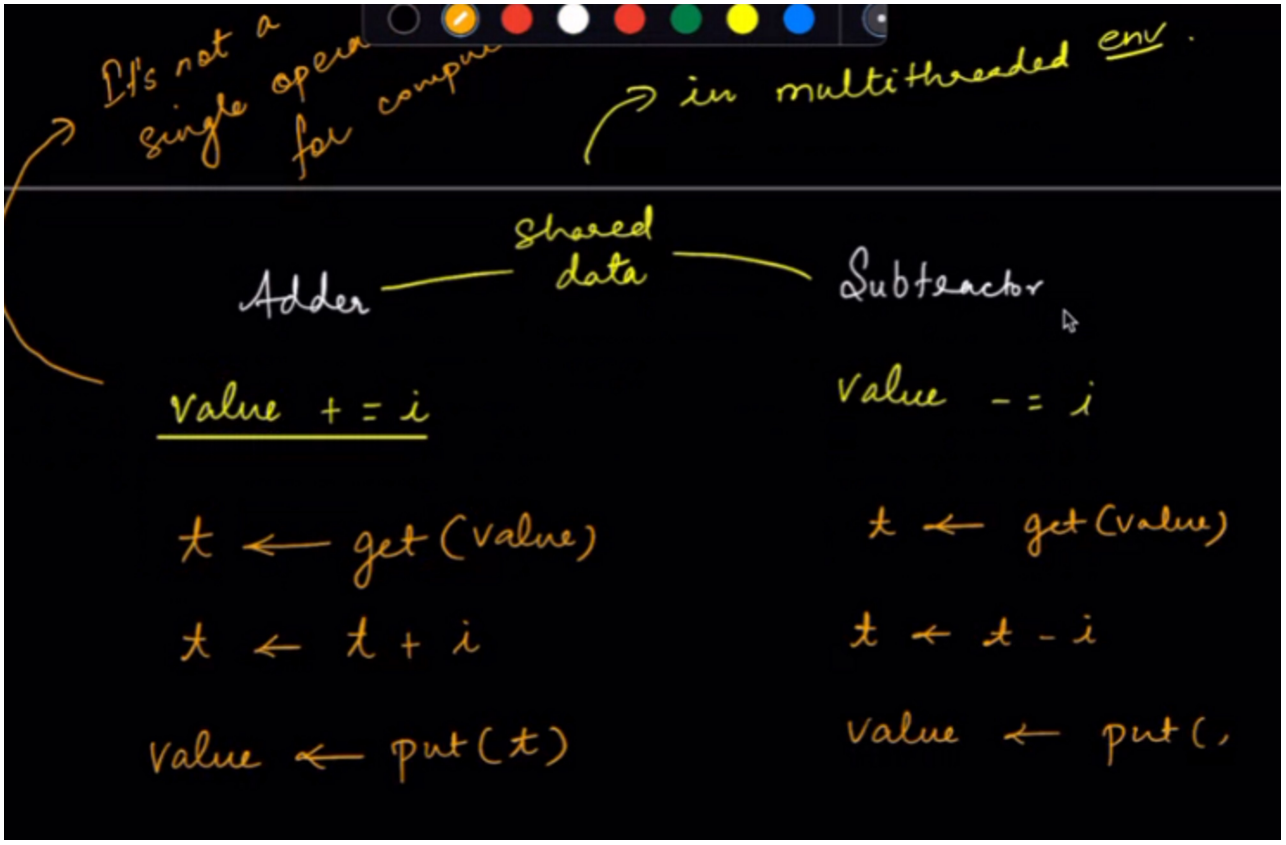


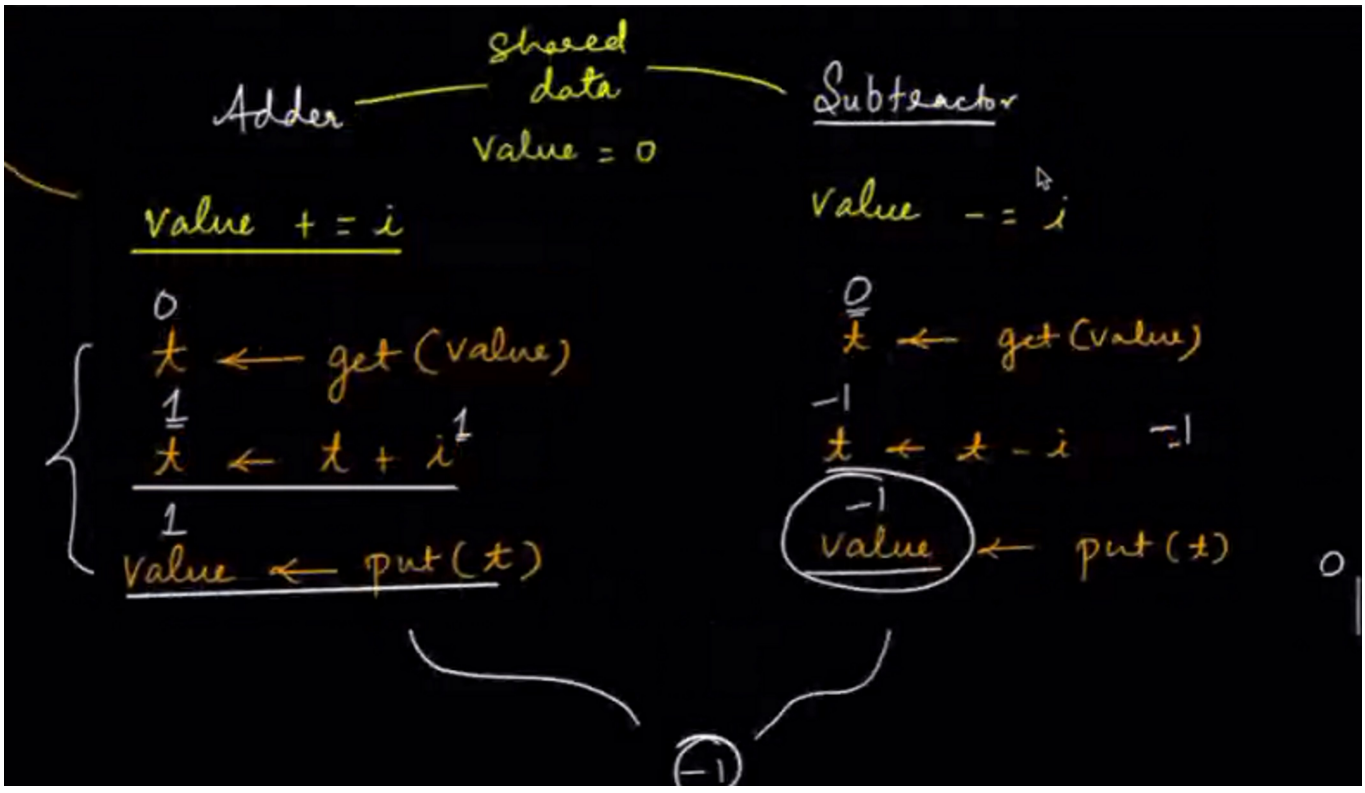
Fix return type to void in adder and subtractor.

Unsynchronised code to implement adder and subtractor using threads.

This code can results in wrong value. So we need to synchronize the code.







https://github.com/learningWithUmang/LLD_May24_Batch/tree/main/src/main/java