

```
import java.util.concurrent.Callable;
import java.util.concurrent.ExecutionException;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
import java.util.concurrent.Future;
import java.util.concurrent.TimeUnit;

class Task implements Callable<String> {
    private int taskId;

    public Task(int taskId) {
        this.taskId = taskId;
    }

    @Override
    public String call() throws Exception {
        System.out.println("Task " + taskId + " started by " + Thread.currentThread().getName());
        Thread.sleep(2000); // simulate work
        return "Task " + taskId + " completed";
    }

    public void setTaskId(int taskId) {
        this.taskId = taskId;
    }
}

public class taskscheduler {

    public static void main(String[] args) {
        // Create fixed thread pool of size 3
    }
}
```

```
ExecutorService executor = Executors.newFixedThreadPool(3);

long startTime = System.currentTimeMillis();

Future<String>[] futures = new Future[5];

// Submit 5 tasks

for (int i = 0; i < 5; i++) {

    futures[i] = executor.submit(new Task(i + 1));

}

// Monitor task execution

for (int i = 0; i < 5; i++) {

    try {

        System.out.println(futures[i].get());

    } catch (InterruptedException | ExecutionException e) {

        System.out.println("Task failed: " + e.getMessage());

    }

}

// Graceful shutdown

executor.shutdown();

try {

    if (!executor.awaitTermination(5, TimeUnit.SECONDS)) {

        executor.shutdownNow();

    }

} catch (InterruptedException e) {

    executor.shutdownNow();

}

long endTime = System.currentTimeMillis();
```

```

        System.out.println("Total Time with ExecutorService: " + (endTime - startTime) + " ms");
    }
}

```

The screenshot shows an IDE interface with the following components:

- EXPLORER** sidebar: Lists files under the SANIKA project, including sorting.py, Student.class, student.ser, studentmanagements..., StudentResultSystem.java, Task.class, taskscheduler.class, taskscheduler.java, ticketbookingsimulation..., TicketCounter.class, UPIPayment.class, userdatastorage.class, userdatastorage.java, users.txt, and userstorage.
- TERMINAL** tab: Displays command-line output from Java running on Microsoft Windows. It includes logs about shared archive files, task execution, and completion times.
- CODE EDITOR**: Shows the source code for the Task class. The code implements Callable<String>, sets a taskId, performs a sleep operation, and returns a completion message. It also has a setTaskId method.

```

Microsoft Windows [Version 10.0.26100.4946]
(c) Microsoft Corporation. All rights reserved.

C:\Users\madha\OneDrive\Desktop\sanika> cmd /C "C:\Users\madha\AppData\Roaming\Code\User\globalStorage\pleiades.java-extension-pack-jdk\java\21\bin\java.exe -XX:+ShowCodeDetailsInExceptionMessages -cp C:\Users\madha\AppData\Roaming\Code\User\workspaceStorage\e8d50f06b353f9154a01ab3563ee4ec2\redhat.java\jdt_ws\sanika_ce92ae38\b in taskscheduler "
[0.023s][warning][cds] This file is not the one used while building the shared archive file: C:\Users\madha\AppData\Roaming\Code\User\globalStorage\pleiades.java-extension-pack-jdk\java\21\lib\modules
[0.026s][warning][cds] C:\Users\madha\AppData\Roaming\Code\User\globalStorage\pleiades.java-extension-pack-jdk\java\21\lib\modules size has changed.
Task 3 started by pool-1-thread-3
Task 1 started by pool-1-thread-1
Task 2 started by pool-1-thread-2
Task 4 started by pool-1-thread-2
Task 5 started by pool-1-thread-3
Task 1 completed
Task 2 completed
Task 3 completed
Task 4 completed
Task 5 completed
Total Time with ExecutorService: 4047 ms

```

```

1 // Source code is decompiled from a .class file using FernFlower decompiler (from IntelliJ)
2 import java.util.concurrent.Callable;
3
4 class Task implements Callable<String> {
5     private int taskId;
6
7     public Task(int var1) {
8         this.taskId = var1;
9     }
10
11    public String call() throws Exception {
12        int var10001 = this.taskId;
13        System.out.println("Task " + var10001 + " started by " + Thread.currentThread().getN
14        Thread.sleep(2000L);
15        return "Task " + this.taskId + " completed";
16    }
17
18    public void setTaskId(int var1) {
19        this.taskId = var1;
20    }
21 }
22

```