

Started on	Tuesday, 17 June 2025, 7:51 PM
State	Finished
Completed on	Tuesday, 17 June 2025, 8:26 PM
Time taken	35 mins 9 secs
Grade	300.00 out of 300.00 (100%)

Question **1**
Correct
Mark 100.00 out of 100.00

Time limit	0.5 s
Memory limit	64 MB


Hitung Prima

Menghitung bilangan prima adalah pekerjaan yang berat secara komputasional. Komputer harus menguji dari 1 sampai bilangan itu sendiri akan angka yang dapat membaginya.

Kalian ditugaskan untuk membuat kelas yang dapat menghitung jumlah prima dalam rentang 1 hingga N (inklusif). Karena angka N akan besar, anda wajib menggunakan concurrency untuk menghindari **Time Limit Exceeded** dan **Runtime Error**. **Jangan lupa untuk mengikuti komen pada file!**

Diberikan file [Hitung.java](#) dan [ThreadTracker.java](#)

Submit kembali file **Hitung.java** saja.

Java 8 

 [Hitung.java](#)

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	100	Accepted	0.35 sec, 36.72 MB

Time limit	1 s
Memory limit	64 MB

Simple Word Count Server

Kalian ditugaskan untuk membuat suatu program simulasi server yang menghitung jumlah kata yang diberikan oleh client. Dalam program ini, setiap request yang datang dari client akan dijalankan dan diproses dalam thread baru.

Server akan menerima kalimat-kalimat yang perlu dihitung jumlah katanya. Kemudian, untuk setiap worker thread yang tersedia pada server akan memproses setiap kalimat tersebut dan pada akhirnya main thread akan mengembalikan jumlah kata pada setiap kalimat.

Penjelasan Soal

- Terdapat kelas `SimpleWordCountServer` memiliki variabel `nWorkers` yang menyatakan maksimum jumlah threads yang tersedia pada server (disebut juga sebagai thread pool).
- Method `processRequests(String[] requests)` akan memetakan setiap elemen pada array requests ke suatu thread secara rata terlebih dahulu. Jika jumlah thread yang tersedia kurang dari jumlah elemen `requests` maka akan terdapat satu atau lebih thread yang mendapatkan elemen lebih. Contohnya jika `nWorkers = 3` dan `requests.length = 4` maka pemetaan elemen:

```
Thread-0 <- requests[0]
Thread-1 <- requests[1]
Thread-2 <- requests[2]
Thread-0 <- requests[3]
```

Terlihat bahwa Thread-0 akan mendapatkan 2 elemen.

Perhatian: Jangan membuat kelas baru untuk membuat suatu worker thread. Gunakanlah deklarasi in-line kelas, seperti berikut

```
Thread thread = new Thread(new Runnable() {
    @Override
    public void run() {
    }
});
```

- Method `countWords(String request)` akan menghitung jumlah kata pada kalimat `request`. Method ini akan dipanggil oleh setiap worker thread.

Files

- Lengkapilah file `SimpleWordCountServer.java`
- Agar tidak membebani server Olympia, gunakanlah file `SimpleWordCountServerDriver.java` untuk pengujian kompilasi dan solusi awal Anda. Pengujian sebenarnya tetap dengan submit ke Olympia.

Submit kembali file `SimpleWordCountServer.java` yang telah berisi jawaban Anda

[attachment.zip](#)

Java 8 ▾

 [SimpleWordCountServer.java](#)

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	12.5	Accepted	0.16 sec, 28.82 MB
2	12.5	Accepted	0.07 sec, 28.00 MB
3	12.5	Accepted	0.09 sec, 32.89 MB
4	12.5	Accepted	0.09 sec, 31.50 MB
5	12.5	Accepted	0.09 sec, 29.50 MB
6	12.5	Accepted	0.09 sec, 29.36 MB
7	12.5	Accepted	0.08 sec, 31.38 MB
8	12.5	Accepted	0.11 sec, 31.21 MB

Time limit	1 s
Memory limit	64 MB

Adventurer Guild

"**The OOPsies**" merupakan salah satu guild petualang yang didirikan seminggu yang lalu oleh Warrior Dar bersama Mage Lia dan Archer Bil. The OOPsies beberapa hari terakhir sedang ramai dibicarakan seantero benua karena berhasil mengalahkan salah satu pengawal raja iblis, dan banyak orang yang mulai bergabung dengan guild ini. setiap anggotanya memiliki wewenang untuk menggunakan uang kas guild bersama. setiap anggota dapat melakukan setoran ke kas guild, dan juga menggunakan uang kas tersebut untuk membeli perlengkapan seperti armor atau senjata. Karena anggotanya yang ramai ini, tiga pendiri guild ingin memastikan bahwa transaksi uang sudah memenuhi prinsip ACID.

Bantu mereka untuk menyusun ***sistem accounting*** yang dapat diandalkan. Lengkapi file [GuildAccount.java](#) lalu kumpulkan dengan nama file yang sama. Untuk membantu debugging dan testing di komputer kalian, disediakan file [Adventurer.java](#) dan [FailedTransactionException.java](#).

Contoh penggunaan:

```

GuildAccount account = new GuildAccount(10000);

Adventurer dar = new Adventurer("Dar");
Adventurer bil = new Adventurer("Bil");
Adventurer lia = new Adventurer("Lia");

// Dar menyetor 2000 gold dan menarik 1500 gold
Thread thread1 = new Thread(() -> {
    try {
        account.deposit(dar, 2000);
        account.withdraw(dar, 1500, "Armor");
    } catch (FailedTransactionException e) {
        // ignore
    }
});

// Bil menarik 3000 gold untuk senjata
Thread thread2 = new Thread(() -> {
    try {
        account.withdraw(bil, 3000, "Sword");
    } catch (FailedTransactionException e) {
        // ignore
    }
});

// Lia menyetor 500 gold dan menarik 2000 gold
Thread thread3 = new Thread(() -> {
    try {
        account.deposit(lia, 500);
        account.withdraw(lia, 2000, "Shield");
    } catch (FailedTransactionException e) {
        // ignore
    }
});

thread1.start();
thread2.start();
thread3.start();

try {
    thread1.join();
    thread2.join();
    thread3.join();
} catch (InterruptedException e) {
    e.printStackTrace();
}

System.out.println("Final balance: " + account.getBalance());
System.out.println("Log:");
System.out.println(account.getLog());
```

output:

```
Final balance: 6000
Log:
Dar men-depositkan uang sejumlah 2000 ke kas. Saldo lama: 10000, Saldo baru: 12000
Dar menarik uang sejumlah 1500 dari kas untuk Armor. Saldo lama: 12000, Saldo baru: 10500
Bil menarik uang sejumlah 3000 dari kas untuk Sword. Saldo lama: 10500, Saldo baru: 7500
Lia men-depositkan uang sejumlah 500 ke kas. Saldo lama: 7500, Saldo baru: 8000
Lia menarik uang sejumlah 2000 dari kas untuk Shield. Saldo lama: 8000, Saldo baru: 6000
```

Java 8

 [GuildAccount.java](#)

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	20	Accepted	0.06 sec, 28.43 MB
2	20	Accepted	0.07 sec, 29.93 MB
3	20	Accepted	0.22 sec, 35.31 MB
4	20	Accepted	0.07 sec, 28.94 MB
5	20	Accepted	0.16 sec, 36.00 MB

[◀ Feedback Form](#)

Jump to...