

Started on	Tuesday, 17 June 2025, 8:28 PM
State	Finished
Completed on	Tuesday, 17 June 2025, 8:37 PM
Time taken	9 mins 14 secs
Grade	300.00 out of 300.00 (100%)

Time limit	1 s
Memory limit	64 MB

Kurir Paket Metapers

REVISI: Ubah SENDIRI parameter HashSet di method validatePacket() menjadi Set (tidak mempengaruhi implementasi, hanya paramnya saja).

Sebelum:

```
public static void validatePacket(List<Node> nodes, Map<String, HashSet<Integer>> transmissionLog, Packet packet) throws
NetworkException {}
```

Setelah revisi:

```
public static void validatePacket(List<Node> nodes, Map<String, Set<Integer>> transmissionLog, Packet packet) throws NetworkException
{} 
```

Tuan Ray adalah seorang **kurir paket jaringan** di dunia Metapers. Sebagai kurir profesional, ia wajib mematuhi protokol komunikasi yang telah ditetapkan oleh **Eppy Foundation**.

Namun, karena tingginya kompleksitas jaringan, Tuan Ray mengalami kesulitan dalam melakukan validasi paket secara manual. Anda, sebagai mahasiswa dari **Institut Teknologi Bombardino (ITB)** yang ahli dalam Java dan jaringan komputer, diminta membantu dengan memodelkan sistem validasi ini.

Aturan Validasi Paket

Sebuah paket dianggap **tidak valid** dan program harus melempar **Exception** jika melanggar salah satu dari aturan berikut:

- Loopback (self-loop):** Paket dikirim ke alamat pengirimnya sendiri. → Throw **LoopbackException**
- Time desync:** Node tujuan sudah pernah menerima paket dengan **timestamp lebih baru atau sama**. → Throw **TimeDesyncException**
- Collision:** Dua atau lebih paket masuk ke **node tujuan yang sama di timestamp yang sama**. → Throw **CollisionException**

Jika semua validasi lolos, maka:

- Timestamp pada **Node.to.lastReceived** diperbarui.
- transmissionLog** untuk node yang sesuai ditambahkan timestamp baru. Jika node belum pernah tercatat, maka buat dulu node-nya sebagai key di Map dengan HashSet kosong, baru tambahkan timestamp.

Contoh Skenario:

Misalkan terdapat tiga node:

Node A, Node B, Node C

Dan terdapat lima paket sebagai berikut:

PacketA = (from A, to A, 5)
PacketB = (from A, to B, 5)
PacketC = (from D, to C, 10)
PacketD = (from B, to C, 10)
PacketE = (from A, to C, 10)
PacketF = (from B, to C, 5)

Validasi dilakukan secara berurutan dengan memanggil:

validatePacket(ListNodes, transmissionLog, packet)

Hasil yang diharapkan:

Paket	Hasil	Alasan
PacketA	LoopbackException	Node pengirim dan penerima sama (A → A)
PacketB	✓ Sukses	B belum pernah menerima paket, valid dan pertama kali menerima waktu 5
PacketC	NetworkException	Node D tidak ada dalam daftar node
PacketD	✓ Sukses	C menerima paket pada timestamp 10
PacketE	CollisionException	C sudah menerima timestamp 10 dari packet D
PacketF	TimeDesyncException	C sudah menerima timestamp 10, sekarang menerima timestamp 5 (mundur)

Tips: Baca hint yang diberikan di soal dengan baik.

Berkas yang diberikan

- [Node.java](#)
- [Packet.java](#)
- [NetworkSimulator.java](#)

Implementasikan dan kumpulkan berkas **NetworkSimulator.java**

Java 8 ▴ ▾

 [NetworkSimulator.java](#)

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	10	Accepted	0.10 sec, 28.93 MB
2	10	Accepted	0.09 sec, 27.89 MB
3	10	Accepted	0.07 sec, 28.83 MB
4	10	Accepted	0.08 sec, 28.71 MB
5	10	Accepted	0.08 sec, 28.26 MB
6	10	Accepted	0.07 sec, 28.52 MB
7	10	Accepted	0.07 sec, 28.27 MB
8	10	Accepted	0.07 sec, 28.94 MB
9	10	Accepted	0.08 sec, 28.14 MB
10	10	Accepted	0.08 sec, 27.94 MB

Time limit	1 s
Memory limit	64 MB

Tuan Bil baru saja masuk menjadi karyawan baru di **Direktorat Pendidikan Intitut Terlalu Bruh (ITB)**! Direktorat ingin mengubah sistem akademik lamanya, yaitu SIX dengan teknologi yang lebih baru. Kamu ditugaskan untuk memodelkan seorang Mahasiswa beserta daftar nilainya untuk sistem baru yang akan dinamakan **SangX**! Sayangnya, Tuan Bil masuk tanpa mengetahui bahasa Java oleh karena itu ia meminta kamu untuk memodelkan kelas tersebut!

Diberikan file [Mahasiswa.java](#) untuk dilengkapi dan disubmit kembali.

Aturan konversi nilai:
Nilai > 90 = Indeks "A"
Nilai > 70 = Indeks "B"
Nilai > 50 = Indeks "C"
Nilai > 40 = Indeks "D"
Sisanya = Indeks "E"

Java 8 ▾

 [Mahasiswa.java](#)

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	10	Accepted	0.07 sec, 28.58 MB
2	10	Accepted	0.06 sec, 28.95 MB
3	10	Accepted	0.06 sec, 29.36 MB
4	10	Accepted	0.06 sec, 28.34 MB
5	10	Accepted	0.06 sec, 29.13 MB
6	10	Accepted	0.07 sec, 28.75 MB
7	10	Accepted	0.06 sec, 27.86 MB
8	10	Accepted	0.06 sec, 30.00 MB
9	10	Accepted	0.06 sec, 26.23 MB
10	10	Accepted	0.06 sec, 26.30 MB

Time limit	1 s
Memory limit	64 MB

Secure Password

Tuan Sid adalah seorang praktisi sekaligus dosen di dunia keamanan siber. Sebagai tugas untuk pekan ini, Tuan Sid meminta kalian untuk menulis program yang dapat menerima sebuah string password lalu memvalidasinya.

Format input:

Berisi string password lengkap **instance** String langsung.

Format output:

1. Apabila validasi password berhasil, maka cetak boolean hasil validasi.
2. Apabila validasi password gagal karena exception, maka cetak pesan dalam format "**<nama exception>!**" diikuti dengan isi message exception tersebut.
 - Program bisa gagal di tahap apapun, tidak hanya pada tahap validasi password saja.
 - Apabila exception yang dikeluarkan merupakan instance Exception selain **InvalidPasswordException** ataupun **InvalidLengthException**, maka cetak nama kelas exception yang dipanggil (bisa dengan getName() saja).
 -
3. Berdasarkan hasil validasi password:
 - Apabila password valid maka cetak "**Password validated.**" pada baris yang berbeda dari output sebelumnya ke layar.
 - Apabila password tidak valid, maka cetak "**Password string error!**" pada baris yang berbeda dari output sebelumnya ke layar.
4. Setelah validasi password selesai (apapun hasilnya), tutup scanner yang dibuka dan cetak "**Operation finished.**" pada baris yang berbeda dari output sebelumnya ke layar.
 - Penutupan scanner bisa dilakukan dengan pemanggilan method close() dari scanner.

Contoh input benar:

```
'''
P@ssw0rd123.
'''
```

Contoh output:

```
'''
true
Password validated.
Operation finished.
'''
```

Contoh input salah (InvalidPasswordException):

```
'''
password1234
'''
```

Contoh output:

```
'''
InvalidPasswordException! Password harus mengandung minimal satu huruf kapital
Password string error!
Operation finished.
'''
```

Contoh input salah (InvalidLengthException):

```
'''
P@ss
'''
```


```
Contoh output:
...
InvalidLengthException! Password harus memiliki panjang minimal 12 karakter
Password string error!
Operation finished.
...
```

Lengkapi [attachment.zip](#) dan submit zip berisi kedua file yang telah dilengkapi tersebut.

Tambahan:

- Fungsi validate() di Password.java melakukan throw Exception dengan ketentuan sebagai berikut:
- Apabila panjang password kurang dari 12, message = "Password harus memiliki panjang minimal 12 karakter"
 - Apabila tidak ada huruf kapital pada password, message = "Password harus mengandung minimal satu huruf kapital"
 - Apabila tidak ada angka pada password, message = "Password harus mengandung minimal satu angka"
 - Apabila tidak ada spesial karakter pada password, message = "Password harus mengandung minimal satu karakter khusus"

Java 8 ▾

 [3.zip](#)

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	10	Accepted	0.08 sec, 28.16 MB
2	10	Accepted	0.06 sec, 27.94 MB
3	10	Accepted	0.06 sec, 28.26 MB
4	10	Accepted	0.06 sec, 27.75 MB
5	10	Accepted	0.06 sec, 27.80 MB
6	10	Accepted	0.06 sec, 28.29 MB
7	10	Accepted	0.07 sec, 29.04 MB
8	10	Accepted	0.06 sec, 27.89 MB
9	10	Accepted	0.06 sec, 28.93 MB
10	10	Accepted	0.06 sec, 26.19 MB

◀ Feedback Form

Jump to... ▾