

Started on	Monday, 5 May 2025, 12:07 AM
State	Finished
Completed on	Monday, 5 May 2025, 12:08 AM
Time taken	1 min 10 secs
Grade	300.00 out of 300.00 (100%)

Time limit	1 s
Memory limit	64 MB

Pasti kalian pernah mencoba membuat fungsi yang dapat menjumlahkan semua bilangan di dalam sebuah list.

```
public static int sum(List<Integer> list) {
    int result = list[0];
    for (int i=1; i<list.size(); i++) {
        result += list[i];
    }
    return result;
}
```

Atau mungkin sebuah fungsi yang dapat menggabungkan (konkatenasi) seluruh string didalam sebuah list.

```
public static String join(List<String> list) {
    String result = list[0];
    for (int i=1; i<list.size(); i++) {
        result += list[i];
    }
    return result;
}
```

Atau bahkan sebuah fungsi yang dapat mencari nilai terbesar dalam sebuah list integer.

```
public static int max(List<Integer> list) {
    int result = list[0];
    for (int i=1; i<list.size(); i++) {
        if (list[i] > result) {
            result = list[i];
        }
    }
    return result;
}
```

Jika dilihat, bentuk ketiga fungsi tersebut sangat serupa. Seketika, Tuan Bil mengingat fungsi-fungsi [Fold](#)), yang salah satu fungsinya adalah, **Reduce**. **Reduce** adalah fungsi yang dapat menggabungkan seluruh elemen dalam List menjadi satu nilai dengan menerapkan operasi fungsi secara berurutan. Karena Tuan Bil sudah lama tidak memegang bahasa ~~Jawa~~ Java, ia meminta kalian untuk membuat fungsi tersebut!

Untuk menggunakan binary operator, dapat menggunakan fungsi apply(x, y) dengan x dan y adalah dua data yang akan dioperasikan. Diberikan file [Fold.java](#) untuk dilengkapi dan disubmit kembali.

Java 8

 [Fold.java](#)

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	7	Accepted	0.15 sec, 33.96 MB
2	7	Accepted	0.15 sec, 33.93 MB
3	7	Accepted	0.14 sec, 33.54 MB

No	Score	Verdict	Description
4	7	Accepted	0.13 sec, 33.73 MB
5	7	Accepted	0.15 sec, 34.05 MB
6	7	Accepted	0.15 sec, 35.77 MB
7	7	Accepted	0.14 sec, 33.39 MB
8	7	Accepted	0.15 sec, 33.25 MB
9	7	Accepted	0.24 sec, 33.42 MB
10	7	Accepted	0.14 sec, 33.49 MB
11	6	Accepted	0.16 sec, 33.86 MB
12	6	Accepted	0.15 sec, 33.77 MB
13	6	Accepted	0.16 sec, 34.00 MB
14	6	Accepted	0.15 sec, 35.75 MB
15	6	Accepted	0.14 sec, 33.66 MB

Time limit	1 s
Memory limit	64 MB

MiniMax PrioQueue

Setelah diminta membuat Deque, kali ini kalian diminta untuk membuat **MiniMax PrioQueue**. MiniMax PrioQueue terdiri dari 3 kelas yaitu:

- **AbstractQueue**: Kelas abstrak yang berisi method-method yang akan digunakan oleh kelas Queue lainnya.
- **MinQueue**: Kelas yang mengimplementasikan queue dengan prioritas minimum.
- **MaxQueue**: Kelas yang mengimplementasikan queue dengan prioritas maksimum.

Kelas-kelas ini akan menggunakan generic type **T** yang merupakan subclass dari **Comparable<T>** sehingga kalian dapat menggunakan fungsi yang disediakan oleh **ArrayList** atau **Collections** untuk mengurutkan elemen-elemen di dalam queue.

Untuk menggunakan method dalam Collections, kalian perlu mengimportnya terlebih dahulu:

```
import java.util.Collections;
```

Berikut method yang dapat digunakan oleh ArrayList:

- **add** : Menambahkan elemen ke dalam list.
- **remove** : Menghapus elemen dari list.
- **get** : Mengambil elemen dari list.

Kumpulkan file **MinQueue.java**, **MaxQueue.java**, dan **AbstractQueue.java** dalam format ZIP dengan nama bebas. Pastikan file ZIP tersebut tidak dalam folder apapun, hanya langsung berisi 3 file tersebut.

Attachments

Attachments: [attachments.zip](#)

Java 8 ▾



[2.zip](#)

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	10	Accepted	0.06 sec, 27.88 MB
2	10	Accepted	0.09 sec, 27.91 MB
3	10	Accepted	0.07 sec, 29.14 MB
4	10	Accepted	0.08 sec, 28.31 MB
5	10	Accepted	0.07 sec, 28.09 MB
6	10	Accepted	0.09 sec, 29.13 MB
7	10	Accepted	0.07 sec, 29.05 MB
8	10	Accepted	0.09 sec, 30.61 MB
9	10	Accepted	0.10 sec, 29.68 MB

No	Score	Verdict	Description
10	10	Accepted	0.10 sec, 29.51 MB

Question **3**

Correct

Mark 100.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

Keranjang

Anda diberikan kelas **Barang** yang memiliki subclass **Baju** dan **Handphone**. Anda diminta mengimplementasikan kelas **Keranjang**, sebuah kelas yang menyimpan tipe data generic untuk kelas-kelas tersebut.


Kelas keranjang akan menyimpan barang (t) yang bertipe T dan tipe dari barang tersebut (type) yang disimpan dalam bentuk String.

Lengkapi metode dan kelas yang ada pada **Keranjang.java**

Hint: gunakan metode getClass().getName() untuk mencari tipe data

attachment : [Keranjang.java](#)

Java 8

 [Keranjang.java](#)

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	20	Accepted	0.09 sec, 29.03 MB
2	20	Accepted	0.07 sec, 27.75 MB
3	20	Accepted	0.06 sec, 27.93 MB
4	20	Accepted	0.06 sec, 28.06 MB
5	20	Accepted	0.07 sec, 28.78 MB

[Feedback Form](#)

Jump to...

[Pra-Praktikum 6 \(belum lengkap karena Aul tidak upload soal\)](#)