Question **1**

Correct

Mark 100.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

Nama File: MenuItem.java, Milkshake.java, NasiGoreng.java

Implementasikan 3 kelas berikut: MenuItem.java, Milkshake.java, NasiGoreng.java

Upload jawaban sebagai berkas **KantinKampus.zip** yang berisi tepat 3 file: MenuItem.java, Milkshake.java, dan NasiGoreng.java

Catatan: Pastikan setiap output diakhiri oleh endline ("\n") atau menggunakan `println`. File - file di atas akan dicompile dalam satu package yang sama, apabila diperlukan, silahkan ubah public-private identifier dari setiap attribut dan method yang ada di dalam kelas - kelas.

Java 8

KantinKampus.zip

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	5	Accepted	0.07 sec, 27.98 MB
2	5	Accepted	0.07 sec, 28.77 MB
3	5	Accepted	0.06 sec, 28.37 MB
4	5	Accepted	0.07 sec, 27.96 MB
5	5	Accepted	0.07 sec, 27.92 MB
6	5	Accepted	0.06 sec, 30.45 MB
7	5	Accepted	0.06 sec, 27.88 MB
8	5	Accepted	0.07 sec, 30.91 MB
9	5	Accepted	0.06 sec, 28.32 MB
10	5	Accepted	0.06 sec, 27.87 MB
11	5	Accepted	0.06 sec, 29.03 MB
12	5	Accepted	0.06 sec, 28.94 MB
13	5	Accepted	0.06 sec, 28.43 MB
14	5	Accepted	0.07 sec, 28.43 MB
15	5	Accepted	0.06 sec, 27.85 MB
16	5	Accepted	0.06 sec, 28.68 MB
17	5	Accepted	0.06 sec, 27.07 MB

Question **2**

Partially correct

Mark 85.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

Nama File: Zooooo.zip

Implementasikan kelas dalam file berikut.

Upload jawaban sebagai berkas **Zooooo.zip** yang berisi tepat 5 file: Animal.java, Aves.java, IAnimal.java, Mammal.java, dan Reptile.java

Catatan: Pastikan setiap output diakhiri oleh endline ("\n") atau menggunakan `println`.

Java 8

Zooooo.zip

Score: 85

Blackbox

Score: 85

Verdict: Wrong answer

Evaluator: Exact

No	Score	Verdict	Description
1	5	Accepted	0.06 sec, 27.86 MB
2	0	Wrong answer	0.06 sec, 28.08 MB
3	5	Accepted	0.06 sec, 27.82 MB
4	5	Accepted	0.06 sec, 28.05 MB
5	5	Accepted	0.06 sec, 27.89 MB
6	5	Accepted	0.06 sec, 28.46 MB
7	5	Accepted	0.06 sec, 27.88 MB
8	5	Accepted	0.06 sec, 28.11 MB
9	5	Accepted	0.07 sec, 30.88 MB
10	5	Accepted	0.06 sec, 28.47 MB
11	5	Accepted	0.06 sec, 28.40 MB
12	0	Wrong answer	0.08 sec, 27.85 MB
13	5	Accepted	0.06 sec, 28.04 MB
14	5	Accepted	0.08 sec, 28.50 MB
15	5	Accepted	0.20 sec, 28.21 MB
16	5	Accepted	0.07 sec, 28.41 MB
17	5	Accepted	0.07 sec, 29.04 MB
18	5	Accepted	0.07 sec, 28.96 MB
19	5	Accepted	0.06 sec, 28.88 MB

Question **3**

Correct

Mark 100.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

Gro, mantan penjahat super, kini telah bangkrut dan memutuskan untuk membuka sebuah toko kecil. Karena keterbatasan dana, Gro tidak mampu membayar pekerja sehingga seluruh operasional toko harus ia kelola sendiri, termasuk menjadi kasir.

Seiring berjalannya waktu, workload yang Gro tangani semakin berat. Untuk meringankan pekerjaannya, Gro berinisiatif membuat sistem kasir digital sederhana. Namun, saat seorang pelanggan ingin membayar menggunakan **E-Wallet**, Gro kebingungan karena sistem yang ia buat hanya mendukung metode **Transfer Bank**.

Gro pun menyadari bahwa sistem yang ia kembangkan tidak fleksibel dan tidak mudah dikembangkan (*scalable*) di masa depan. Oleh karena itu, bantulah Gro membenahi sistem kasirnya dengan menerapkan **konsep OOP (Object-Oriented Programming)**, khususnya **inheritance**, sebagaimana yang telah Anda pelajari di kelas.

Buatlah sebuah program dalam bahasa Java dengan spesifikasi sebagai berikut:

1. **Kelas Abstrak:** `Pembayaran`
   - Atribut (gunakan modifier yang hanya mengizinkan attribut dapat diakses oleh objek kelasnya dan *subclassnya*):
     - `String namaPelanggan`
     - `double jumlah`
   - Method abstrak:
     - `double hitungTotal();` → menghitung total yang harus dibayar pelanggan.
   - Method tambahan:
     - `String getNamaPelanggan();` → mengembalikan nilai `namaPelanggan`.
2. **Subclass** `COD` (Cash on Delivery)
   - Meng-override method `hitungTotal()`.
   - Rumus: `jumlah + 5000`.
3. **Subclass** `EWallet`
   - Meng-override method `hitungTotal()`.
   - Rumus: `jumlah + (jumlah * 0.01)`.
4. **Subclass** `TransferBank`
   - Meng-override method `hitungTotal()`.
   - Rumus: `jumlah + 4000`.

Kumpulkan semua kelas ke dalam Cashier.zip. (Note: Nama file sesuaikan dengan nama kelas nya)

Java 8

Cashier.zip

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	10	Accepted	0.06 sec, 28.05 MB

Question **4**

Correct

Mark 100.00 out of 100.00

Time limit	1 s
Memory limit	64 MB

Nama File: OperatingSystem.java, Linux.java, MacOS.java, Windows.java, UsageType.java

Buat sistem penilaian kompatibilitas untuk berbagai *Operating System* dengan menerapkan konsep **inheritance** pada Java. Implementasikan keseluruhan struktur sesuai ketentuan berikut.

Ketentuan Umum

- Seluruh atribut pada setiap kelas harus diberi modifier `private` dan nilainya dapat diakses dengan fungsi getter, contoh: getBaseScore. Penamaan fungsi getter adalah get<Nama Attribute> dengan nama fungsi tersebut dituliskan dalam Camel Case.
- Method `printInfo` dan `calculateCompatibility` wajib `public`.
- Method `calculateCompatibility` dihitung dengan cara baseScore + multiplier * baseScore
- Method bantu pada kelas abstrak, yaitu `clampScore` dan `getAdditionalInfo`, harus menggunakan modifier `protected`.
- Enum, konstruktor, dan getter yang disebutkan pada spesifikasi harus `public`.

1. Kelas Abstrak OperatingSystem

- Atribut: `name` (String), `version` (String), `kernelType` (String), `baseScore` (double 0–100).
- Buat konstruktor dengan dengan parameter name, version, dan kernelType, dan baseScore. Apabila parameter yang diberikan bernilai null, maka akan digantikan dengan default value yaitu secara berurutan: "Unknown OS", "Unknown Version", "Unknown Kernel", dan 0.
- Buat method abstrak `double calculateCompatibility(UsageType usage)`.
- Sediakan method bantu `protected double clampScore(double score)` untuk memastikan skor tetap di rentang 0–100.
- Implementasikan method `void printInfo(UsageType usage)` untuk menampilkan informasi OS beserta nilai kompatibilitas hasil `calculateCompatibility`.
- **Format Input**:
  - Konstruktor menerima parameter `name`, `version`, `kernelType`, dan `baseScore` dalam urutan tersebut. Setiap nilai teks null atau kosong harus diganti dengan default yang sudah ditentukan.
  - Method `calculateCompatibility` dan `printInfo` menerima satu argumen bertipe `UsageType` yang menunjukkan skenario penggunaan yang sedang dinilai.
- **Format Output**:
  - `printInfo` harus mencetak lima baris utama secara berurutan:

    ```
    Name: <name>
    Version: <version>
    Kernel Type: <kernelType>
    {Additional Information jika ada.}
    Base Score: <baseScore dengan 2 digit desimal>
    Compatibility for <usage.name()>: <skor kompatibilitas dengan 2 digit desimal>.
    ```

  - Jika `getAdditionalInfo()` mengembalikan string non-kosong, baris tersebut dicetak setelah informasi kernel dan sebelum baris `Base Score`.
  - Gunakan `System.out.printf` atau formatter lain yang setara sehingga nilai numerik muncul dengan dua angka di belakang koma dan garis baru di akhir setiap baris.

2. Enum UsageType

- Buat enum `UsageType` dengan nilai `SERVER`, `DEVELOPMENT`, dan `GAMING`.

3. Kelas Turunan

- **Linux**
  - Tambahkan enum internal `Distro` dengan nilai `UBUNTU`, `FEDORA`, `ARCH`, `DEBIAN`.
  - Atribut tambahan: `distroType` (Distro). Jika null, gunakan default `UBUNTU`.
  - Aturan multiplier terhadap `baseScore`:
    - `SERVER` → +20%
    - `DEVELOPMENT` → +10%
    - `GAMING` → -15%
  - **Format Input**:
    - Konstruktor menerima parameter tambahan `distroType`. Jika null, setel ke `Distro.UBUNTU`.
    - `getDistroType()` mengembalikan nilai enum untuk kebutuhan pengujian.
  - **Format Output**:
    - `getAdditionalInfo()` harus mengembalikan string `"Distribution: " + distroType.name()`.
    - `printInfo` wajib menampilkan baris tambahan `Distribution: <DISTRO>` tepat sebelum baris `Base Score`, sesuai nilai yang dikembalikan oleh `getAdditionalInfo`.

- **MacOS**
  - Atribut tambahan: `hasMSeriesChip` (boolean).
  - Aturan multiplier baseScore:
    - `DEVELOPMENT` → +25%
    - `GAMING` → -20%
    - Jika `hasMSeriesChip == true` → +10%
  - **Format Input**:
    - Konstruktor menerima parameter tambahan boolean `hasMSeriesChip` yang menentukan apakah chip seri-M tersedia.
    - Sediakan method `hasMSeriesChip()` bertipe boolean untuk membaca nilai atribut tersebut.
  - **Format Output**:
    - `getAdditionalInfo()` harus mengembalikan string `"Has M-Series Chip: " + (hasMSeriesChip ? "Yes" : "No")`.
    - `printInfo` wajib menampilkan baris `Has M-Series Chip: Yes/No` sesuai nilai atribut sebelum baris `Base Score`.
- **Windows**
  - Enum internal `Edition` dengan nilai `HOME`, `PRO`, `SERVER`.
  - Atribut tambahan: `edition` (Edition). Jika null, gunakan default `HOME`.
  - Aturan multiplier baseScore:
    - Jika `edition == SERVER` dan penggunaan `SERVER` → +30%
    - Untuk penggunaan `GAMING` → +20%
    - Jika `edition == HOME` dan penggunaan `SERVER` → -5%
  - **Format Input**:
    - Konstruktor menerima parameter tambahan `edition`; gunakan `Edition.HOME` apabila nilainya null.
    - Method `getEdition()` harus mengembalikan nilai enum untuk validasi.
  - **Format Output**:
    - `getAdditionalInfo()` harus mengembalikan string `"Edition: " + edition.name()`.
    - `printInfo` wajib menampilkan baris `Edition: <EDITION>` sesuai dengan nilai enum sebelum menampilkan `Base Score`.

Pastikan seluruh skor kompatibilitas akhir dijaga menggunakan `clampScore` agar tidak melebihi 100 atau turun di bawah 0.

Format Output Tambahan

- Method `calculateCompatibility` pada setiap kelas harus mengembalikan nilai akhir bertipe `double` yang sudah diproses oleh `clampScore` sehingga berada di rentang 0–100.

Catatan Tambahan

- Gunakan `println` agar setiap baris output berakhir dengan `\n`.
- Seluruh kelas harus berada di paket default (tanpa deklarasi `package`).
- Gunakan file Main.java untuk melakukan pengujian kelas - kelas yang telah kalian buat.

Upload jawaban sebagai berkas **operating-system.zip** yang berisi keenam file di atas.

Java 8

operating-system.zip

Score: 100

Blackbox

Score: 100

Verdict: Accepted

Evaluator: Exact

No	Score	Verdict	Description
1	25	Accepted	0.27 sec, 29.68 MB
2	25	Accepted	0.33 sec, 28.50 MB
3	25	Accepted	0.29 sec, 28.11 MB
4	25	Accepted	0.30 sec, 28.36 MB

◄ Pasca Praktikum 2