NO 1

Buatlah sebuah program robot sederhana yang dapat menerima perintah untuk melakukan aksi. Robot dapat menolak perintah apabila perintah yang diberikan tidak sesuai dengan daftar perintah yang dia ketahui.

Lengkapi dan submit file `Robot.java`.

File tersebut akan memiliki sebuah kelas, yaitu `Robot`.

Kelas Robot memiliki atribut sebagai berikut:

- Atribut `x` bertipe `integer` yang melambangkan lokasi robot di sumbu X dengan nilai awal 0.
- Atribut `y` bertipe `integer` yang melambangkan lokasi robot di sumbu Y dengan nilai awal 0.

Kelas Robot memiliki 2 buah method sebagai berikut:

- walk yang menerima 2 buah parameter, yaitu `x` bertipe `integer` dan `y` bertipe `integer`.
- talk yang menerima sebuah parameter, yaitu `language` bertipe `string`.
- receiveCommand yang menerima sebuah parameter, yaitu `command` bertipe `string`.

Penjelasan masing-masing method sebagai berikut:

Pada method `walk`, robot menerima parameter `x` dan `y` dan menambahkan nilai itu ke atribut `x` dan `y` yang dimilikinya. Robot lalu akan mengeprint pesan berupa `Sedang berjalan menuju (x, y)` dimana `x` dan `y` adalah nilai akhir setelah penambahan. Apabila nilai `x` atau `y` setelah ditambahkan akan melebihi 10, method ini melemparkan sebuah `exception` dengan pesan `Tidak bisa jalan`. Nilai `x` dan `y` tidak ditambahkan pada kasus ini.

Robot hanya dapat berbicara menggunakan bahasa Indonesia dan bahasa Inggris saja. Oleh karena itu, pada pemanggilan method `talk`, apabila parameter `language` bukan `Indonesia` atau `Inggris`, robot akan melemparkan sebuah `exception` dengan pesan `Tidak bisa berbicara dalam bahasa {language}`. Apabila input valid, robot akan mengeprint pesan berupa `Sedang berbicara dalam bahasa {language}`.

Method `receiveCommand` digunakan untuk mengecek apakah input `command` benar, input `command` yang valid adalah `walk` dan `talk`. Selain kedua command tersebut, method akan melemparkan exception dengan pesan `Perintah {command} tidak dikenal`.

NO 2

Buatlah sebuah program kalkulator yang bisa menerima kalkulasi sederhana. Meskipun sederhana, kalkulator akan menerapkan konsep Exception dimana kalkulator akan mengeluarkan pesan error ketika input tidak benar.

Lengkapi dan submit file `Calculator.java`.

File tersebut akan memiliki 3 buah kelas, yaitu `Calculator`, `InvalidOperationException` yang mengextend kelas Exception, dan `InvalidDivisionByZero` yang mengextend kelas Exception. `InvalidOperationException` akan menerima input berupa pesan custom ketika diinisiasi. Sedangkan, `InvalidDivisionByZero` akan mengembalikan string mutlak ketika pembagian 0.

Kelas `Calculator.java` yang berisi fungsi calculate yang mereturn `double`.

Fungsi tersebut akan menerima 3 buah parameter, yaitu `a` bertipe `double` yang berupa inputan pertama, `b` bertipe `double` yang berupa inputan kedua, dan `c` bertipe `String` berupa operator yang akan di eksekusi oleh kalkulator. (inputan pertama dan kedua berarti a-b, a+b, a*b, dan a/b)

Pertama, Kalkulator akan memeriksa jenis operasi yang dimasukkan pada parameter. Apabila operasi yang diinput pengguna bukan `+`, `-`, `*`, atau `/`, maka kalkulator akan mengembalikan pesan error dari kelas `InvalidOperationException.java` yang berisi pesan dengan format `"Invalid operation: " + operation;`.

Apabila kalkulator menerima input pembagian terhadap 0, maka kalkulator juag akan mengembalikan pesan error dari kelas `InvalidDivisionByZero.java` yang berisi pesan dengan format `Tidak dapat melakukan pembagian terhadap 0`.

NO 3

Exception - Calculator (Driver)

Menggunakan `Calculator.java` yang sudah dibuat di soal sebelumnya, buatlah program utama yang menerima perintah untuk melakukan kalkulasi sederhana.

Format input:

1. Baris pertama berisi input angka pertama.
2. Baris kedua berisi input angka kedua.
3. Baris ketiga berisi operasi kalkulasi apa yang akan dilakukan oleh angka kedua kepada angka pertama.

Format output:

1. Apabila kalkulasi berhasil, maka cetak angka hasil kalkulasi.
2. Apabila kalkulasi gagal/program gagal karena *exception*, maka cetak pesan dalam format `<nama exception>!` diikuti dengan isi message exception tersebut.
    1. Program bisa gagal di tahap apapun, tidak hanya pada tahap kalkulasi saja.
    2. Apabila `exception` yang dikeluarkan merupakan instance *Exception* selain `InvalidOperationException` ataupun `InvalidDivisionByZero`, maka cetak `Unknown exception!`
3. Setelah operasi kalkulasi selesai (apapun hasilnya), tutup scanner yang dibuka dan cetak `Calculated.` ke layar.
    1. Penutupan scanner bisa dilakukan dengan pemanggilan method `close()` dari scanner.

Contoh input **benar**:

20.0

4.0

*

Contoh output:

80.0

Calculated.

Contoh input **salah** (InvalidOperationException):

20.0

4.0

#

Contoh output:

InvalidOperationException! Invalid operation: #

Calculated.

Contoh input **salah** (Unknown Exception):

a

Contoh output:

Unknown exception!

Calculated.

Lengkapi dan submit file Main.java.

JAWABAN:

NO 1

```
/**
 * Jangan lupa tambahkan kata kunci untuk menghandle exception
 */
public class Robot {
    /**
     * Tambahkan atribut kelas disini
     */
    private int xPos;
    private int yPos;
```

```
public Robot() {
    /**
     * Konstruktor
     * Jangan lupa inisialisasi atribut kelas
     */
    this.xPos = 0;
    this.yPos = 0;
}

public void walk(int x, int y) throws Exception{
    /**
     * Implementasi
     * Apabila hasil penambahan x atau y melebihi 10, dilempar sebuah exception
     * dengan pesan "Tidak bisa jalan". Atribut tidak ditambahkan pada kasus ini.
     *
     * Apabila input valid, tambahkan nilai atribut dan print pesan
     * "Sedang berjalan menuju (x, y)"
     */
    int newXPos = xPos + x;
    int newYPos = yPos + y;
    if (newXPos > 10 || newYPos > 10) {
        throw new Exception("Tidak bisa jalan");
    }
    else {
        xPos = newXPos;
```

```
        yPos = newYPos;
        System.out.println("Sedang berjalan menuju (" + xPos + ", " + yPos + ")");
    }
}

public void talk(String language) throws Exception{
    /**
     * Implementasi
     * Apabila input bukan "Indonesia" atau "Inggris", lempar exception dengan pesan
     * "Tidak bisa berbicara dalam bahasa {language}"
     *
     * Apabila input valid, print pesan "Sedang berbicara dalam bahasa {language}"
     */
    if (!language.equals("Indonesia") && !language.equals("Inggris")) {
        throw new Exception("Tidak bisa berbicara dalam bahasa " + language);
    }
    else {
        System.out.println("Sedang berbicara dalam bahasa " + language);
    }
}

public void receiveCommand(String command) throws Exception{
    /**
     * Implementasi
     * Apabila input bukan "walk" atau "talk", lempar exception "Perintah {command} tidak dikenal"
     */
```

```
        if (!command.equals("walk") && !command.equals("talk")) {
            throw new Exception("Perintah " + command + " tidak dikenal");
        }
    }
}
```