Melakukan Operasi Matriks di Python

Matriks dapat dikatakan sebagai *list* dua dimensi dimana suatu *list* berisi *list* lagi. Untuk merepresentasikan matriks, kita harus menyimpan *list* dengan panjang yang sama dalam suatu *list*. Bila *list* berbeda - beda panjangnya, maka *list* tersebut disebut sebagai *sparse matrix*. Sebagai contoh berikut adalah contoh representasi matriks di Python:

```
matrix = [
    [5, 0],
    [2, 6],
]

matrix1 = [
    [5, 0, 8],
    [2, 6, 7],
    [1, 3, 4],
]

matrix2 = [
    [1, 0, 0, 0],
    [0, 1, 0, 0],
    [0, 0, 1, 0],
    [0, 0, 0, 1],
]
```

Dalam mengolah matriks, ada berbagai operasi yang dapat dilakukan. Mulai dari translasi, rotasi, mencari determinan, operasi baris elementer, dan lainnya. Namun kita hanya akan membahas beberapa operasi dasar seperti penjumlahan, pengurangan dan perkalian dua matriks.

Mengakses Elemen Matriks

Setelah mendefinisikan suatu matriks, tentu kita ingin mengakses setiap elemen pada matriks tersebut. Dengan memanfaatkan *nested loop* **for**, kita dapat mengakses suatu elemen pada koordinat tertentu.

Sebagai contoh mari kita lihat *source code* yang menampilkan isi matriks berikut:

```
matrix = [
    [5, 0],
    [2, 6],
]

for x in range(0, len(matrix)):
    for y in range(0, len(matrix[0])):
        print (matrix[x][y], end=' '),
    print
```

Bila kita eksekusi di konsol maka akan muncul *output* berikut:

```
$ python3 simple-matrix.py
5 0
2 6
```

Melakukan Penjumlahan Matriks

Penjumlahan matriks dilakukan dengan menjumlahkan setiap elemen. Hasil penjumlahan tersebut akan menjadi elemen baru. Masing - masing matriks kita akses setiap elemennya pada koordinat yang sama kemudian kita jumlahkan untuk mendapatkan elemen baru

```
mat1 = [
    [5, 0],
    [2, 6],
]

mat2 = [
    [1, 0],
    [4, 2],
]


for x in range(0, len(mat1)):
    for y in range(0, len(mat1[0])):
        print (mat1[x][y] + mat2[x][y], end=' '),
    print
```

Jika *source code* kita jalankan, maka akan muncul *output* seperti berikut:

```
$ python3 add-matrix.py
6 0
6 8
```

Melakukan Pengurangan Matriks

Tidak berbeda jauh dengan penjumlahan matriks, pada pengurangan matriks kita hanya mengganti operatornya saja dengan tanda kurang (-). Maka matriks baru akan terbentuk sebagai hasil dari pengurangan setiap kedua elemen matriks. Sebagai contoh berikut adalah *source code* untuk melakukan pengurangan matriks:

```
mat1 = [
    [5, 0],
    [2, 6],
]


mat2 = [
    [1, 0],
    [4, 2],
]

for x in range(0, len(mat1)):
    for y in range(0, len(mat1[0])):
        print (mat1[x][y] - mat2[x][y], end=' '),
    print
```

Jika *source code* kita jalankan, maka akan muncul *output* seperti berikut:

```
$ python3 substract-matrix.py
4 0
-2 4
```

Melakukan Perkalian Matriks

Perkalian matriks merupakan salah satu operasi dasar yang *tricky*. Karena di dalamnya bukan hanya terdapat operasi perkalian, melainkan juga penjumlahan. Perkalian suatu matriks memang tidak sama dengan bilangan biasa, tidak juga langsung mengalikan setiap elemen. Perkalian matriks dilakukan dengan menjumlahkan hasil perkalian suatu baris matriks pertama ke kolom matriks kedua. Setiap baris di matriks pertama akan dikalikan ke setiap kolom di matriks kedua.

Di Python, kita akan menggunakan *nested loop* **for** di dalam **nested loop** yang kedua. *Looping* ketiga tersebut kita gunakan untuk melakukan proses penjumlahan hasil perkalian baris dan kolom. Hasilnya elemen matriks baru akan ditempatkan pada koordinat tersebut. Sebagai contoh berikut adalah *source code* yang melakukan proses perkalian matriks:

```
mat1 = [
    [5, 0],
    [2, 6],
]

mat2 = [
    [1, 0],
    [4, 2],
]

mat3 = []

for x in range(0, len(mat1)):
    row = []
    for y in range(0, len(mat1[0])):
        total = 0
        for z in range(0, len(mat1)):
            total = total + (mat1[x][z] * mat2[z][y])
        row.append(total)
    mat3.append(row)


for x in range(0, len(mat3)):
    for y in range(0, len(mat3[0])):
        print (mat3[x][y], end=' ')
    print ()
```