# Pertemuan 6: Multi-dimensional Array

**Problem 1**

Diketahui tiga buah array `arr1`, `arr2` dan `arr3` berdimensi 2 dan bertipe integer dengan ukuran 3x3

a. Isilah array `arr1` dan `arr2` dengan nilai sembarang (buatlah algoritma inisialisasi array)
b. Lakukan operasi penjumlahan pada `arr1` dan `arr2` dan simpan nilainya ke dalam `arr3`
   Contoh:

**arr3**

| | 1 | 2 | 3 |
|---|---|---|---|
| 1 | 7 | 4 | 5 |
| 2 | 2 | 7 | 9 |
| 3 | 4 | 11 | 9 |

=

**arr1**

| | 1 | 2 | 3 |
|---|---|---|---|
| 1 | 6 | 2 | 3 |
| 2 | 0 | 4 | 8 |
| 3 | 3 | 9 | 6 |

+

**arr2**

| | 1 | 2 | 3 |
|---|---|---|---|
| 1 | 1 | 2 | 2 |
| 2 | 2 | 3 | 1 |
| 3 | 1 | 2 | 3 |

c. Tampilkan nilai `arr3` ke layar sesuai dengan baris dan kolom pada matrks dan setiap nilai dipisahkan dengan spasi.

*Jawaban :*

**Kamus Data**

```
Array of array of Integer: arr1, arr2, arr2
Integer                  : baris, kolom
```

**Algoritma**

```
1.  BEGIN
    |  {initialize arr1}
2.  |  baris <- 1
3.  |  WHILE baris <= 3
4.  |  |  kolom <- 1
5.  |  |  WHILE kolom <= 3
6.  |  |  |  arr1[baris, kolom] <- READ(Keyboard)
7.  |  |  |  kolom <- kolom + 1
8.  |  |  ENDWHILE
9.  |  |  baris <- baris + 1
10. |  ENDWHILE
```

```
    |  {initialize arr2}
11. |  baris <- 1
12. |  WHILE baris <= 3
13. |  |  kolom <- 1
14. |  |  WHILE kolom <= 3
15. |  |  |  arr2[baris, kolom] <- READ(Keyboard)
16. |  |  |  kolom <- kolom + 1
17. |  |  ENDWHILE
18. |  |  baris <- baris + 1
19. |  ENDWHILE
    |  {operate and displaying arr3}
20. |  baris <- 1
21. |  WHILE baris <= 3
22. |  |  kolom <- 1
23. |  |  WHILE kolom <= 3
24. |  |  |  arr3[baris, kolom] <- arr1[baris, kolom] +
    |  |  |  arr2[baris, kolom]
25. |  |  |  WRITE(Screen) arr3[baris, kolom], " "
26. |  |  |  kolom <- kolom + 1
27. |  |  ENDWHILE
28. |  |  WRITE_NEWLINE()
29. |  |  baris <- baris + 1
30. |  ENDWHILE
31. END
```

**Tracing**

**arr1**

```
Iterasi  : 1          2          3          4
counter  : 1  2  3  4  1  2  3  4  1  2  3  4
assign   : 3  5  0     0  6  1     0  2  5
```

$$\begin{bmatrix} 3 & 5 & 0 \\ 0 & 6 & 1 \\ 0 & 2 & 5 \end{bmatrix}$$

**Arr2**

```
Iterasi  : 1          2          3          4
counter  : 1  2  3  4  1  2  3  4  1  2  3  4
assign   : 0  1  0     9  2  0     0  5  1
```

$$\begin{bmatrix} 0 & 1 & 0 \\ 9 & 2 & 0 \\ 0 & 5 & 1 \end{bmatrix}$$

**arr3**

```
Iterasi  :  1               2               3               4
counter  :  1    2    3    4    1    2    3    4    1    2    3    4
assign   : 3+0  5+1  0+0     0+9  6+2  1+0     0+0  2+5  5+1
```

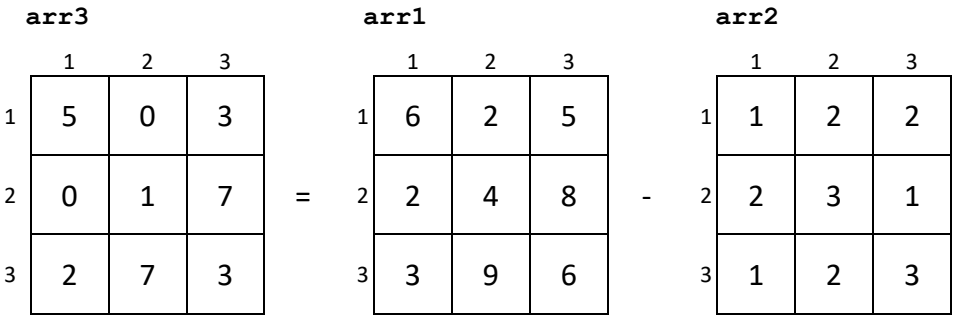
**Output**

```
3 6 0
9 8 1
0 7 6
```

**Problem 2**

Diketahui tiga buah array `arr1`, `arr2` dan `arr3` berdimensi 2 dan bertipe integer dengan ukuran 3x3

a. Isilah array `arr1` dan `arr2` dengan nilai sembarang (buatlah algoritma inisialisasi array)
b. Lakukan operasi pengurangan pada `arr1` dan `arr2` dan simpan nilainya ke dalam `arr3`
   Contoh:




c. Tampilkan nilai `arr3` ke layar sesuai dengan baris dan kolom pada matrks dan setiap nilai dipisahkan dengan spasi.

*Jawaban :*

**Kamus Data**

```
Array of array of Integer: arr1, arr2, arr2
Integer                  : baris, kolom
```

**Algoritma**

```
1.  BEGIN
    |  {initialize arr1}
2.  |  baris <- 1
3.  |  WHILE baris <= 3
4.  |  |  kolom <- 1
5.  |  |  WHILE kolom <= 3
6.  |  |  |  arr1[baris, kolom] <- READ(Keyboard)
7.  |  |  |  kolom <- kolom + 1
8.  |  |  ENDWHILE
9.  |  |  baris <- baris + 1
10. |  ENDWHILE
    |  {initialize arr2}
11. |  baris <- 1
12. |  WHILE baris <= 3
13. |  |  kolom <- 1
14. |  |  WHILE kolom <= 3
```

```
15. |  |  |  arr2[baris, kolom] <- READ(Keyboard)
16. |  |  |  kolom <- kolom + 1
17. |  |  ENDWHILE
18. |  |  baris <- baris + 1
19. |  ENDWHILE
    |  {operate and displaying arr3 }
20. |  baris <- 1
21. |  WHILE baris <= 3
22. |  |  kolom <- 1
23. |  |  WHILE kolom <= 3
24. |  |  |  arr3[baris, kolom] <- arr1[baris, kolom] -
    |  |  |  arr2[baris, kolom]
25. |  |  |  WRITE(Screen) arr3[baris, kolom], " "
26. |  |  |  kolom <- kolom + 1
27. |  |  ENDWHILE
28. |  |  WRITE_NEWLINE()
29. |  |  baris <- baris + 1
30. |  ENDWHILE
31. END
```

**Tracing**

**arr1**

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Iterasi | : 1 | | | | ~~2~~ | | | | ~~3~~ | | | | 4 |
| counter | : 1 | ~~2~~ | ~~3~~ | ~~4~~ | ~~1~~ | ~~2~~ | ~~3~~ | ~~4~~ | ~~1~~ | ~~2~~ | ~~3~~ | ~~4~~ | |
| assign | : 3 | 5 | 0 | | 0 | 6 | 1 | | 0 | 2 | 5 | | |

$$\begin{bmatrix} 3 & 5 & 0 \\ 0 & 6 & 1 \\ 0 & 2 & 5 \end{bmatrix}$$

**Arr2**

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Iterasi | : 1 | | | | ~~2~~ | | | | ~~3~~ | | | | 4 |
| counter | : 1 | ~~2~~ | ~~3~~ | ~~4~~ | ~~1~~ | ~~2~~ | ~~3~~ | ~~4~~ | ~~1~~ | ~~2~~ | ~~3~~ | ~~4~~ | |
| assign | : 0 | 1 | 0 | | 9 | 2 | 0 | | 0 | 5 | 1 | | |

$$\begin{bmatrix} 0 & 1 & 0 \\ 9 & 2 & 0 \\ 0 & 5 & 1 \end{bmatrix}$$

**arr3**

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Iterasi | : 1 | | | | | | ~~2~~ | | | | | | ~~3~~ | | | | 4 |
| counter | : 1 | | ~~2~~ | | ~~3~~ | ~~4~~ | ~~1~~ | | ~~2~~ | | ~~3~~ | ~~4~~ | ~~1~~ | | ~~2~~ | ~~3~~ | ~~4~~ |
| assign | : 3-0 | | 5-1 | | 0-0 | | 0-9 | | 6-2 | | 1-0 | | 0-0 | | 2-5 | 5-1 | |

**Output**

```
3 4 0
-9 4 1
0 -3 4
```

**Problem 3**

Diketahui sebuah array `arr1` berdimensi 2, bertipe integer dan berukuran 5x5

a. Isilah array dengan `arr1` nilai 0

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 | 0 |

b. Ubahlah nilai elemen pada `arr1` pada posisi baris dan kolom sama dengan angka 1. Contoh: `arr1[1,1] <- 1`

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 0 | 0 |
| 2 | 0 | 1 | 0 | 0 | 0 |
| 3 | 0 | 0 | 1 | 0 | 0 |
| 4 | 0 | 0 | 0 | 1 | 0 |
| 5 | 0 | 0 | 0 | 0 | 1 |

c. Ubahlah nilai elemen `arr1` sedemikian rupa sehingga berisi -1 pada diagonal kiri ke kanan

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 0 | -1 |
| 2 | 0 | 1 | 0 | -1 | 0 |
| 3 | 0 | 0 | 1 | 0 | 0 |
| 4 | 0 | -1 | 0 | 1 | 0 |
| 5 | -1 | 0 | 0 | 0 | 1 |

d. Ubahlah nilai elemen `arr1` sisanya dengan angka 5

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 1 | 5 | 5 | 5 | -1 |
| 2 | 5 | 1 | 5 | -1 | 5 |
| 3 | 5 | 5 | 1 | 5 | 5 |
| 4 | 5 | -1 | 5 | 1 | 5 |
| 5 | -1 | 5 | 5 | 5 | 1 |

*Jawaban :*

**Kamus Data**

```
Array of array of integer: arr
Integer                  : baris, kolom
```

**Algoritma**

```
1.  BEGIN
    |  {a. initialize the array with a value of 0}
2.  |  baris <- 1
3.  |  WHILE baris <= 5
4.  |  |  kolom <- 1
5.  |  |  WHILE kolom <= 5
6.  |  |  |  arr[baris, kolom] <- 0
7.  |  |  |  kolom <- kolom + 1
8.  |  |  ENDWHILE
9.  |  |  baris <- baris + 1
10. |  ENDWHILE
    |  {b. reassign 1 to the diagonal}
11. |  baris <- 1
12. |  WHILE baris <= 5
13. |  |  arr[baris, kolom] <- 1
14. |  |  baris <- baris + 1
15. |  ENDWHILE
    |  {c. reassign -1 to another diagonal}
16. |  baris <- 1
17. |  kolom <- 5
18. |  WHILE baris <= 5
19. |  IF baris != 3 THEN
```

```
20. |  |  arr[baris, kolom] <- -1
21. |  ENDIF
22. |  baris <- baris + 1
23. |  kolom <- kolom - 1
24. |  ENDWHILE
    |  {d. change every value of 0 of the array to 5}}
25. |  baris <- 1
26. |  WHILE baris <= 5
27. |  |  kolom <- 1
28. |  |  WHILE kolom <= 5
29. |  |  |  IF arr[baris, kolom] != 0 THEN
30. |  |  |     arr[baris, kolom] <- 5
31. |  |  |  ENDIF
32. |  |  |  kolom <- kolom + 1
33. |  |  ENDWHILE
34. |  |  baris <- baris + 1
35. |  ENDWHILE
36. END
```

**Tracing**

*{a. initialize the array with a value of 0}*

| baris | kolom | index | assign |
|---|---|---|---|
| ~~1~~ | ~~1~~ | [1, 1] | 0 |
| | ~~2~~ | [1, 2] | 0 |
| | ~~3~~ | [1, 3] | 0 |
| | ~~4~~ | [1, 4] | 0 |
| | ~~5~~ | [1, 5] | 0 |
| | ~~6~~ | | |
| ~~2~~ | ~~1~~ | [2, 1] | 0 |
| | ~~2~~ | [2, 2] | 0 |
| | ~~3~~ | [2, 3] | 0 |
| | ~~4~~ | [2, 4] | 0 |
| | ~~5~~ | [2, 5] | 0 |
| | ~~6~~ | | |
| ~~3~~ | ~~1~~ | [3, 1] | 0 |
| | ~~2~~ | [3, 2] | 0 |
| | ~~3~~ | [3, 3] | 0 |
| | ~~4~~ | [3, 4] | 0 |
| | ~~5~~ | [3, 5] | 0 |
| | ~~6~~ | | |

| baris | kolom | index | assign |
|---|---|---|---|
| ~~4~~ | ~~1~~ | [4, 1] | 0 |
| | ~~2~~ | [4, 2] | 0 |
| | ~~3~~ | [4, 3] | 0 |
| | ~~4~~ | [4, 4] | 0 |
| | ~~5~~ | [4, 5] | 0 |
| | ~~6~~ | | |
| ~~5~~ | ~~1~~ | [5, 1] | 0 |
| | ~~2~~ | [5, 2] | 0 |
| | ~~3~~ | [5, 3] | 0 |
| | ~~4~~ | [5, 4] | 0 |
| | ~~5~~ | [5, 5] | 0 |
| | 6 | | |
| 6 | | | |

result:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

*{b. reassign 1 to the diagonal}*

| baris | index | assign |
|---|---|---|
| ~~1~~ | [1, 1] | 1 |
| ~~2~~ | [2, 2] | 1 |
| ~~3~~ | [3, 3] | 1 |
| ~~4~~ | [4, 4] | 1 |
| ~~5~~ | [5, 5] | 1 |
| ~~6~~ | | |

result

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

*{c. reassign -1 to another diagonal}*

| baris | kolom | index | assign |
|---|---|---|---|
| ~~1~~ | ~~5~~ | [1, 5] | -1 |
| ~~2~~ | ~~4~~ | [2, 4] | -1 |
| ~~3~~ | ~~3~~ | [3, 3] | |
| ~~4~~ | ~~2~~ | [4, 2] | -1 |
| ~~5~~ | ~~1~~ | [5, 1] | -1 |
| ~~6~~ | ~~0~~ | | |

result

$$\begin{bmatrix} 1 & 0 & 0 & 0 & -1 \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

*{d. change every value of 0 of the array to 5}*

| baris | kolom | index | assign |
|---|---|---|---|
| ~~1~~ | ~~1~~ | [1, 1] | |
| | ~~2~~ | [1, 2] | 5 |
| | ~~3~~ | [1, 3] | 5 |
| | ~~4~~ | [1, 4] | 5 |
| | ~~5~~ | [1, 5] | |
| | ~~6~~ | | |
| ~~2~~ | ~~1~~ | [2, 1] | 5 |
| | ~~2~~ | [2, 2] | |
| | ~~3~~ | [2, 3] | 5 |
| | ~~4~~ | [2, 4] | |
| | ~~5~~ | [2, 5] | 5 |
| | ~~6~~ | | |
| ~~3~~ | ~~1~~ | [3, 1] | 5 |
| | ~~2~~ | [3, 2] | 5 |
| | ~~3~~ | [3, 3] | |
| | ~~4~~ | [3, 4] | 5 |
| | ~~5~~ | [3, 5] | 5 |
| | ~~6~~ | | |

| baris | kolom | index | assign |
|---|---|---|---|
| ~~4~~ | ~~1~~ | [4, 1] | 5 |
| | ~~2~~ | [4, 2] | |
| | ~~3~~ | [4, 3] | 5 |
| | ~~4~~ | [4, 4] | |
| | ~~5~~ | [4, 5] | 5 |
| | ~~6~~ | | |
| ~~5~~ | ~~1~~ | [5, 1] | |
| | ~~2~~ | [5, 2] | 5 |
| | ~~3~~ | [5, 3] | 5 |
| | ~~4~~ | [5, 4] | 5 |
| | ~~5~~ | [5, 5] | |
| | 6 | | |
| 6 | | | |

**Final result**

$$\begin{bmatrix} 1 & 5 & 5 & 5 & -1 \\ 5 & 1 & 5 & -1 & 5 \\ 5 & 5 & 1 & 5 & 5 \\ 5 & -1 & 0 & 1 & 5 \\ -1 & 5 & 5 & 5 & 1 \end{bmatrix}$$