```
> 
> #Arrays
> 
> a := Array(3);
```
$a := Array(\{() = 3\}, datatype = anything, storage = rectangular, order = Fortran\_order)$ **(1)**

```
> a;
```
$Array(\{() = 3\}, datatype = anything, storage = rectangular, order = Fortran\_order)$ **(2)**

```
> for num from 1 to 3 do a[num] := num; end do;
Error, number of indices exceeds rank
> a[1];
Error, number of indices exceeds rank
> ?Array
> a := Array([seq(i, i = 1 ..10)]);
```
$$a := \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \end{bmatrix} \quad \textbf{(3)}$$

```
> a[6];
```
$$6 \quad \textbf{(4)}$$

```
> #we can initialize arrays with lists
> a := Array( );
```
$a := Array(\{\}, datatype = anything, storage = rectangular, order = Fortran\_order)$ **(5)**

```
> a := [seq(i, i = 1 ..10)];
```
$$a := [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] \quad \textbf{(6)}$$

```
> 
> #Arrays can be printed
> print(a);
```
$$[1, 2, 3, 4, 5, 6, 7, 8, 9, 10] \quad \textbf{(7)}$$

```
> 
> #the standard way to allocate and initialize arrays
> P := Array(1 ..10, 1 ..10, 0);
```
$$P := \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \textbf{(8)}$$

```
> #Assigning values to array elements
> for i from 1 to 10 do
```

```
for j from 1 to 10 do
P[i, j] := 1;
end do;
end do;
```

```
> 
> print(P);
```

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{9}$$

```
> 
> #row major numbering
> k := 0;
for i from 1 to 10 do
for j from 1 to 10 do
k := k + 1;
P[j, i] := k;
end do;
end do;
```

$$k := 0 \tag{10}$$

```
> P
```

$$\begin{bmatrix} 1 & 11 & 21 & 31 & 41 & 51 & 61 & 71 & 81 & 91 \\ 2 & 12 & 22 & 32 & 42 & 52 & 62 & 72 & 82 & 92 \\ 3 & 13 & 23 & 33 & 43 & 53 & 63 & 73 & 83 & 93 \\ 4 & 14 & 24 & 34 & 44 & 54 & 64 & 74 & 84 & 94 \\ 5 & 15 & 25 & 35 & 45 & 55 & 65 & 75 & 85 & 95 \\ 6 & 16 & 26 & 36 & 46 & 56 & 66 & 76 & 86 & 96 \\ 7 & 17 & 27 & 37 & 47 & 57 & 67 & 77 & 87 & 97 \\ 8 & 18 & 28 & 38 & 48 & 58 & 68 & 78 & 88 & 98 \\ 9 & 19 & 29 & 39 & 49 & 59 & 69 & 79 & 89 & 99 \\ 10 & 20 & 30 & 40 & 50 & 60 & 70 & 80 & 90 & 100 \end{bmatrix} \tag{11}$$

```
> # Generating a pattern
> P := Array(1..10, 1..10, (i, j) → piecewise(i ≤ 2 or i ≥ 9 or j ≤ 2 or j ≥ 9, 1, _));
```

$$P := \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & \_ & \_ & \_ & \_ & \_ & \_ & 1 & 1 \\ 1 & 1 & \_ & \_ & \_ & \_ & \_ & \_ & 1 & 1 \\ 1 & 1 & \_ & \_ & \_ & \_ & \_ & \_ & 1 & 1 \\ 1 & 1 & \_ & \_ & \_ & \_ & \_ & \_ & 1 & 1 \\ 1 & 1 & \_ & \_ & \_ & \_ & \_ & \_ & 1 & 1 \\ 1 & 1 & \_ & \_ & \_ & \_ & \_ & \_ & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \quad \textbf{(12)}$$

```
> print(P);
```

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & \_ & \_ & \_ & \_ & \_ & \_ & 1 & 1 \\ 1 & 1 & \_ & \_ & \_ & \_ & \_ & \_ & 1 & 1 \\ 1 & 1 & \_ & \_ & \_ & \_ & \_ & \_ & 1 & 1 \\ 1 & 1 & \_ & \_ & \_ & \_ & \_ & \_ & 1 & 1 \\ 1 & 1 & \_ & \_ & \_ & \_ & \_ & \_ & 1 & 1 \\ 1 & 1 & \_ & \_ & \_ & \_ & \_ & \_ & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \quad \textbf{(13)}$$

```
>
> #Transposing an array (exchanging rows and columns)
> A := Array(1..3, 1..3, (i,j)->3*(i-1)+j);
```

$$A := \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \quad \textbf{(14)}$$

```
> #One correct implementation
> for i from 1 to 3 do
  for j from 1 to 3 do
  if i > j then
  tmp := A[i,j];
  A[i,j] := A[j,i];
  A[j,i] := tmp;
  end if;
  end do;
  end do;
> A
```

$$\begin{bmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{bmatrix} \quad (15)$$

```
>
> #A slightly more efficient implementation
> for i from 1 to 3 do
  for j from i to 3 do
  tmp := A[i,j];
  A[i,j] := A[j,i];
  A[j,i] := tmp;
  end do;
  end do;
> A
```

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \quad (16)$$

```
>
>
```