```
> with(ArrayTools)
```

[AddAlongDimension, Alias, AllNonZero, AnyNonZeros, Append, BlockCopy, CircularShift, ComplexAsFloat, Compress, Concatenate, Copy, DataTranspose, Diagonal, Dimensions, ElementDivide, ElementMultiply, ElementPower, Extend, Fill, FlipDimension, GeneralInnerProduct, GeneralOuterProduct, HasNonZero, HasZero, Insert, IsEqual, IsMonotonic, IsZero, Lookup, LowerTriangle, MultiplyAlongDimension, NumElems, Partition, Permute, PermuteInverse, RandomArray, ReduceAlongDimension, RegularArray, Remove, RemoveSingletonDimensions, Replicate, Reshape, Reverse, ScanAlongDimension, SearchArray, Size, SuggestedDatatype, SuggestedOrder, SuggestedSubtype, Uncompress, UpperTriangle] **(1)**

```
> # Exercice 1
> fact_it := proc(n)
  local i, N;
  N := 1;
  for i from 1 to n do
  N := i·N;
  od;
  N;
  end:
> fact_it(10)
```

3628800 **(2)**

```
> fact_rec := proc(n);
  if n = 0 then 1 else n·fact_rec(n − 1) fi;
  end:
> fact_rec(10)
```

3628800 **(3)**

```
> # Exercice 2
> fibo := proc(n);
  if (n = 0 or n = 1) then 1 else fibo(n − 1) + fibo(n − 2) fi;
  end:
> seq(fibo(k), k = 1..10)
```

1, 2, 3, 5, 8, 13, 21, 34, 55, 89 **(4)**

```
> # Exercice 3
> expo := proc(a, n)
  local b, q, r;
  if n = 0 then 1 else
  q, r := iquo(n, 2), irem(n, 2);
  b := expo(a, q);
  if r = 0 then b^2 else a·b^2 fi;fi;
  end:
> seq(expo(2, k), k = 0..10)
```

1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 **(5)**

```
> T := Array(1..10);for i from 1 to 10 do T[i] := i od: T
```

$$T := \left[\begin{array}{cccccccccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right]$$

$$\left[\begin{array}{cccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \end{array}\right] \quad (6)$$

```
> T[1..7]; T[-1] # On peut facilement accéder à une tranche d'un tableau, et au dernier élément
```

$$\left[\begin{array}{ccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array}\right]$$

$$10 \quad (7)$$

```
> # Exercice 4
> insert := proc(T, j)
  local a;
  # On suppose que T[1..j-1] est trié, puis on insère T[j] à la bonne place dans cette tranche
  if j = 1 then T:
  else if T[j] < T[j - 1] then
  a := T[j - 1];
  T[j - 1] := T[j];
  T[j] := a;
  insert(T, j - 1)
  fi;
  fi;
  end:

> init_rand := proc(n)
  local T, i;
  T := Array(1..n);
  for i from 1 to n do
  T[i] := rand(0..100)( )
  od;
  T
  end:
> T := init_rand(10)
```

$$T := \left[\begin{array}{cccccccccc} 89 & 2 & 48 & 48 & 66 & 100 & 100 & 73 & 89 & 73 \end{array}\right] \quad (8)$$

```
> insert(T, 2)
```

$$\left[\begin{array}{cccccccccc} 2 & 89 & 48 & 48 & 66 & 100 & 100 & 73 & 89 & 73 \end{array}\right] \quad (9)$$

```
> tri_insert := proc(T)
  local j, taille;
  taille := Size(T)[2];
  for j from 2 to taille do insert(T, j) od ; T;end:
>  T := init_rand(10); tri_insert(T)
```

$$T := \left[\begin{array}{cccccccccc} 84 & 8 & 67 & 82 & 62 & 99 & 69 & 35 & 35 & 9 \end{array}\right]$$

$$\left[\begin{array}{cccccccccc} 8 & 9 & 35 & 35 & 62 & 67 & 69 & 82 & 84 & 99 \end{array}\right] \quad (10)$$

```
> # Exercice 5
> dicho := proc(T, x)
  local taille, m, k;
  taille := Size(T)[2];
```

```
    m := floor( taille / 2 );
    if taille = 1 then
    return 0 :
    else if T[m] = x then
    return m :
     else if T[m] < x then
    k := dicho(T(m + 1 ..taille), x);
    if k = 0 then
    return 0
    else
    return m + k
    fi;
   else dicho(T(1 ..m), x);
    fi;
    fi;
    fi;
   end:
```

```
> T := init_rand(10); tri_insert(T)
```

$$T := \left[\begin{array}{cccccccccc} 66 & 58 & 65 & 11 & 48 & 100 & 89 & 34 & 14 & 25 \end{array}\right]$$

$$\left[\begin{array}{cccccccccc} 11 & 14 & 25 & 34 & 48 & 58 & 65 & 66 & 89 & 100 \end{array}\right] \quad \textbf{(11)}$$

```
> dicho(T, 52); dicho(T, 25); dicho(T, 66)
```

$$0$$

$$3$$

$$8 \quad \textbf{(12)}$$

```
> restart; with(ArrayTools)
```

[*AddAlongDimension*, *Alias*, *AllNonZero*, *AnyNonZeros*, *Append*, *BlockCopy*, *CircularShift*, *ComplexAsFloat*, *Compress*, *Concatenate*, *Copy*, *DataTranspose*, *Diagonal*, *Dimensions*, *ElementDivide*, *ElementMultiply*, *ElementPower*, *Extend*, *Fill*, *FlipDimension*, *GeneralInnerProduct*, *GeneralOuterProduct*, *HasNonZero*, *HasZero*, *Insert*, *IsEqual*, *IsMonotonic*, *IsZero*, *Lookup*, *LowerTriangle*, *MultiplyAlongDimension*, *NumElems*, *Partition*, *Permute*, *PermuteInverse*, *RandomArray*, *ReduceAlongDimension*, *RegularArray*, *Remove*, *RemoveSingletonDimensions*, *Replicate*, *Reshape*, *Reverse*, *ScanAlongDimension*, *SearchArray*, *Size*, *SuggestedDatatype*, *SuggestedOrder*, *SuggestedSubtype*, *Uncompress*, *UpperTriangle*] **(13)**

```
> # Tri fusion
> fusion :=proc(T1, T2)
   local n1, n2, T, c, i, j, k;
   n1 := Size(T1)[2]; n2 := Size(T2)[2];
   T := Array(1 ..n1 + n2);
  c := 1; i := 1; j := 1;
   while i ≤ n1 and j ≤ n2 do
   if T1[i] < T2[j] then
```

```
    T[c] := T1[i];
    i := i + 1;
    else
    T[c] := T2[j];
    j := j + 1;
    fi;
    c := c + 1
    od;
    if i = n1 + 1 then
    for k from j to n2 do
    T[c] := T2[k];
    c := c + 1
    od;
    else
    for k from i to n1 do
    T[c] := T1[k];
    c := c + 1
    od;
    fi;
    return T;
    end:
```

```
> tri_fusion := proc(T)
    local taille, m;
    taille := Size(T)[2];
    if taille ≤ 1 then
    return T;
    else
    m := floor(taille/2);
    fusion(tri_fusion(T(1..m)), tri_fusion(T(m + 1..taille)))
    fi;
    end:
```

```
> T1 := init_rand(10); tri_insert(T1); T2 := init_rand(10); tri_insert(T2)
```

$$T1 := \left[\begin{array}{cccccccccc} 82 & 24 & 89 & 92 & 59 & 92 & 13 & 49 & 46 & 7 \end{array}\right]$$

$$\left[\begin{array}{cccccccccc} 7 & 13 & 24 & 46 & 49 & 59 & 82 & 89 & 92 & 92 \end{array}\right]$$

$$T2 := \left[\begin{array}{cccccccccc} 45 & 43 & 8 & 76 & 58 & 15 & 0 & 69 & 76 & 38 \end{array}\right]$$

$$\left[\begin{array}{cccccccccc} 0 & 8 & 15 & 38 & 43 & 45 & 58 & 69 & 76 & 76 \end{array}\right] \quad \textbf{(14)}$$

```
> fusion(T1, T2)
```

$$\left[\begin{array}{cccccccccccccccccccc} 0 & 7 & 8 & 13 & 15 & 24 & 38 & 43 & 45 & 46 & 49 & 58 & 59 & 69 & 76 & 76 & 82 & 89 & 92 & 92 \end{array}\right] \quad \textbf{(15)}$$

```
> T := init_rand(10); tri_fusion(T)
```

$$T := \left[\begin{array}{cccccccccc} 17 & 42 & 54 & 39 & 16 & 69 & 51 & 80 & 86 & 33 \end{array}\right]$$

$$\left[\begin{array}{cccccccccc} 16 & 17 & 33 & 39 & 42 & 51 & 54 & 69 & 80 & 86 \end{array}\right] \tag{16}$$

```
>
>
```