

Task 1

Create a function `MultiplyAndSkip` that accepts numbers α and ω . Your function needs to return the first 7 integers multiplied by α and then added to ω :

```
1 MultiplyAndSkip 0
1 2 3 4 5 6 7
2 MultiplyAndSkip 0
2 4 6 8 10 12 14
2 MultiplyAndSkip -3
-1 1 3 5 7 9 11
3 MultiplyAndSkip 1
4 7 10 13 16 19 22
```

Task 2

Create functions `mat1`, `mat2`, and `mat3` that ignore their argument and always return the following matrices:

```
mat1 0
```

```
1 2 3
```

```
4 5 6
```

```
mat2 0
```

```
-1 0 1
```

```
2 3 4
```

```
mat3 0
```

```
-1 -2
```

```
-3 -4
```

```
-5 -6
```

Task 3

Create two functions, VC and HC, that accept matrices and behave as follows:

```
      mat1 VC mat2
1  2  3
4  5  6
-1 0  1
2  3  4
```

```
      HC mat3
-1 -2 -1 -2 -1 -2
-3 -4 -3 -4 -3 -4
-5 -6 -5 -6 -5 -6
```

Task 4

Create a function `ToVector` that accepts any array as ω and turns it into a vector with the same elements:

```

    ToVector 1 2 3 4
1 2 3 4
    ToVector 3 2pi6
1 2 3 4 5 6
    ToVector 2 2 2pi8
1 2 3 4 5 6 7 8
```

Task 5

Create a function `Sum2ndLast` that accepts a numeric array as w and sums its contents along the 2nd to last axis:

```
Sum2ndLast 2 3pi6
5 7 9
Sum2ndLast 2 10 2pi40
100 110
300 310
Sum2ndLast 2 2 10 2pi80
100 110
300 310

500 510
700 710
```

Task 6

Write a function `DropRandRows` that accepts a matrix as ω and drops, from the top, between 1 and all the rows of the matrix (chosen randomly).

Here are some example runs:

```
DropRandRows 3 4p112
```

```
5  6  7  8
```

```
9 10 11 12
```

```
DropRandRows 3 4p112
```

A Empty result!

```
DropRandRows 3 4p112
```

```
9 10 11 12
```

```
DropRandRows 3 4p112
```

```
9 10 11 12
```

Task 7

Write a function `PickRandCols` that accepts a matrix as ω and extracts between 1 and all columns from the left (chosen randomly).

Here are some example runs:

```
PickRandCols 2 5\r10
```

```
1 2 3
```

```
6 7 8
```

```
PickRandCols 2 5\r10
```

```
1
```

```
6
```

```
PickRandCols 2 5\r10
```

```
1 2 3 4
```

```
6 7 8 9
```

Task 8

Write a function `RIota` that takes a positive integer ω and produces the same result as $\iota \omega$, but in the reverse order:

```
      RIota 5
5 4 3 2 1
      RIota 1
1
      RIota 10
10 9 8 7 6 5 4 3 2 1
```


Task 9

Using RIoT a, write a function `Reverse` that takes a vector w and returns the same vector in the reverse order:

```
Reverse 15
5 4 3 2 1
Reverse 'Hello, world!'
!dlrow ,olleH
Reverse 15 30 2 8
8 2 30 15
```

Task 10

Create a function `ColumnsSurpass` that accepts a number as α and a matrix as ω , and returns all columns of ω whose sum is strictly greater than α :

```
40 ColumnsSurpass 4 5\r20
3  4  5
8  9 10
13 14 15
18 19 20
48 ColumnsSurpass 4 5\r20
5
10
15
20
```

Task 11

Define a function `Range` that accepts a numeric vector ω and returns the amplitude of ω , that is, returns the difference between the largest element of ω and the smallest element of ω :

```
9      Range 1 10
100    Range 0 100
100    Range 100 0
128    Range 15 64 23 -64 -15
```

Task 12

Create a function `MaxRangeRow` that accepts a numeric matrix w and returns the row with the largest range:

```
MaxRangeRow 2 3\r0 10 5 1 2 3  
0 10 5  
MaxRangeRow 2 3\r1 2 3 10 0 5  
10 0 5  
MaxRangeRow 5|2 3\r6  
4 0 1
```

Task 13

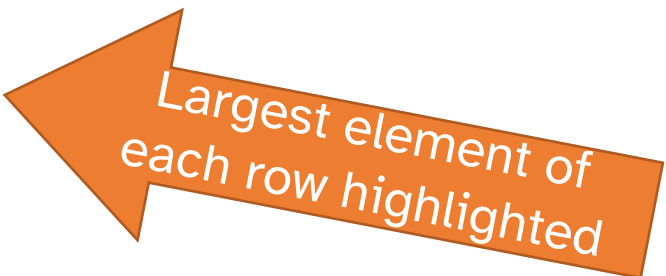
Create a function `IsInteger` that takes a numeric vector ω and returns 1 if ω is an integer, and 0 otherwise:

```
IsInteger 3
1
IsInteger 3.3
0
IsInteger ^3 3 3.3
1 1 0
IsInteger (1:10)%2
0 1 0 1 0 1 0 1 0 1
```

Task 14

Write a function `TopDown` that accepts a numeric matrix w and reorders its rows so that the row with the largest number shows up first and the row with the smallest largest number shows at the bottom:

| | | TopDown | 4 | 3 | 0 | 12 | 9 | 8 | 1 | 2 | 3 | 4 | 10 | 5 | 6 | 7 |
|---|---|---------|---|----|---|----|---|---|---|---|---|---|----|---|---|---|
| 0 | | 12 | | | | | 9 | | | | | | | | | |
| 3 | | 4 | | 10 | | | | | | | | | | | | |
| | 8 | | 1 | | 2 | | | | | | | | | | | |
| 5 | | 6 | | 7 | | | | | | | | | | | | |
| | | TopDown | 3 | 3 | 1 | 9 | | | | | | | | | | |
| 7 | 8 | 9 | | | | | | | | | | | | | | |
| 4 | 5 | 6 | | | | | | | | | | | | | | |
| 1 | 2 | 3 | | | | | | | | | | | | | | |



Largest element of each row highlighted

Task 15

Write a function `RemoveFrom` that accepts a vector α and removes those items from the vector ω :

```
'abc' RemoveFrom 'cabana'
n
'abc' RemoveFrom 'cape'
pe
1 2 3 RemoveFrom 0 1 0 2 0 45
0 0 0 45
```

Task 16 (bonus)

Write a function `RemoveExtra` that accepts a vector w and returns the same vector after removing the occurrences of its most common element:

```
RemoveExtra 'banana'
bnn
RemoveExtra 2|17
0 0 0
RemoveExtra 'Mississippi'
Miiippi
```


Task 17

Write the function `TimesTable` to generate the multiplication table for numbers up to w :

`TimesTable 3`

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

`TimesTable 5`

| | | | | |
|---|----|----|----|----|
| 1 | 2 | 3 | 4 | 5 |
| 2 | 4 | 6 | 8 | 10 |
| 3 | 6 | 9 | 12 | 15 |
| 4 | 8 | 12 | 16 | 20 |
| 5 | 10 | 15 | 20 | 25 |

Task 18

Write the function `IsSorted` which determines if a given array is already in ascending order:

```
IsSorted 3 1 4 1 5
0

IsSorted 2 3 5 7 11
1

IsSorted 3 4p 'Bob Abe Carl '
0

IsSorted 3 4p 'Abe Bob Carl '
1
```

Task 19

Write the function `AnyCopies` which determines if the vector w has any duplicates:

```
AnyCopies 'India'
0
AnyCopies 'Indian'  # there are 2 "n"s
1
```

Task 20

Write the function `Trim` which removes leading spaces from the vector `w`:

```
Trim '      poof '
poof
Trim '    I have spaces '
I have spaces
Trim 'nospace'
nospace
```

Task 21

Write the function `KeepOnly` which removes from the vector α any element not in ω :

```
'Hello World' KeepOnly "A"
```

```
HW
```

```
3 1 4 1 5 KeepOnly 13
```

```
3 1 1
```

Task 22

Write the function `HasElements` which determines if the array w has any elements (i.e. it isn't empty):

| | |
|---|--|
| | <code>HasElements 3 1 4</code> |
| 1 | |
| | <code>HasElements 1 0</code> |
| 0 | |
| | <code>HasElements 3 2 4</code> \square A |
| 1 | |
| | <code>HasElements 3 0 4</code> \square A |
| 0 | |
| | <code>HasElements 4 2</code> |
| 1 | |

Task 23

Write the function `Overlaps` which determines if the arrays α and ω have any elements in common.

```
2 7 1 8 Overlaps 3 1 4
```

```
1
```

```
1 2 3 Overlaps 7 8 9 10
```

```
0
```

Task 24

Write the function `OfLength` which takes a matrix α and returns the rows that have exactly w letters. Remove any spaces from the result.

```
names←7 6p'Patel Arya Babu Dewan Singh GandhiGupta '
```

```
names OfLength 5
```

```
Patel
```

```
Dewan
```

```
Singh
```

```
Gupta
```

```
pnames OfLength 5
```

```
4 5
```

```
names OfLength 6
```

```
Gandhi
```

```
pnames OfLength 6
```

```
1 6
```


Task 25

Write the function `Explode` which generates a vector consisting of one 1, two 2s, three 3s, until the argument number:

`Explode 5`

1 2 2 3 3 3 4 4 4 4 5 5 5 5 5

`Explode 3`

1 2 2 3 3 3

`Explode 1`

1

`Explode 0`

0

Task 26

Write the function `NoFizzBuzz` which removes any elements from the vector `w` which are divisible by 3 or 5.

```
NoFizzBuzz 1:10
1 2 4 7 8
NoFizzBuzz 2 4 6 8 10 12
2 4 8
NoFizzBuzz 2 4 6 8 -10 12    a don't forget negatives!
2 4 8
```

Task 27 (bonus)

Write the function `CentreIn` which takes a character vector α and adds spaces on the left and right so it becomes w characters long, with the original text approximately centred. The number of added spaces on the left and right must not differ by more than 1.

```
'Boo' CentreIn 5
```

```
 Boo 
```

```
ρ 'Boo' CentreIn 5
```

```
5
```

```
'Boom' CentreIn 5    a returning 'Boom' is also OK
```

```
 Boom
```

```
'Hi' CentreIn 10
```

```
 Hi 
```