## Task 1

1. `)CLEAR` to clear your workspace

2. `)SAVE` your workspace with a workspace ID like **your_name_tasks5.dws**

## Task 2

Write a function `Upper` to convert a word into upper case.

```
      Upper 'apl'
APL
      Upper 'works'
WORKS
```

## Task 3

Write a function `Clean` that changes all non-digits into stars:

```
      Clean 'Easy as 1, 2 and 3'
********1**2*****3
      Clean '1000'
1000
      Clean 'APL works!'
**********
```

## Task 4 (bonus task)

Rewrite `Upper` to convert any character vector into upper case, even if the text contains spaces and punctuation:

```
      Upper 'apl works!'
APL WORKS!
      Upper 'works'
WORKS
```

# Task 5

Define a function `CountVowels` to count the number of vowels in the character vector `⍵`

```
      CountVowels 'AeiOU'
5
      CountVowels 'Mississippi'
4
      CountVowels 'We have TWELVE vowels in this sentence.'
12
```

## Task 6

Define a function `RowEquals` to locate the vector ⍺ in the matrix ⍵

```
      text ← 3 5⍴'GREATGIANTTIGER'
      'TIGER' RowEquals text
0 0 1
      'GREAT' RowEquals text
1 0 0

      fruits←3 7⍴'OrangesMangoesBananas'
      'Bananas' RowEquals fruits
0 0 1
      'Carrots' RowEquals fruits
0 0 0
```

# Task 7

Define a function `Up` that accepts a vector of numbers and sorts them in increasing order:

```
      Up 0  ¯3  1.5  10  4.2
¯3  0  1.5  4.2  10
      Up 5 ¯2 0 2
¯2 0 2 5
      Up ¯4 1 ¯3 ¯3 1 3 1 2 0 3
¯4 ¯3 ¯3 0 1 1 1 2 3 3
```

Define a function `Down` that sorts the vector of numbers in decreasing order:

```
      Down 0  ¯3  1.5  10  4.2
10  4.2  1.5  0  ¯3
      Down 5 ¯2 0 2
5 2 0 ¯2
      Down ¯4 1 ¯3 ¯3 1 3 1 2 0 3
3 3 2 1 1 1 0 ¯3 ¯3 ¯4
```

## Task 8

Define a function `SortedBy` that accepts a character vector (representing name initials) on the left and a vector of numbers (ages) on the right. Reorder the initials so that the youngest person comes first, and the oldest person comes last.

```
      'ABCDE' SortedBy 20 24 83 18 35
'DABEC'
      1 5 5 3 2 8 SortedBy 'ABCYXZ'
1 5 5 2 3 8
      1 5 5 3 2 8 SortedBy 'CBAYXZ'
5 5 1 2 3 8
      'AABCBBA' SortedBy 10 5 12 13 5 3 6
BABAABC
       'ABBCDBA' SortedBy ⌽⍳7
ABDCBBA
```

## Task 9

Define a function `MatrixSortedDownBy` that takes a character matrix on the left and a numeric vector on the right. It should reorder the rows of the matrix so that the row corresponding to the highest number comes first.

```
      products ← 5 6⍴'coffeebread curry beans milk  '
      products MatrixSortedDownBy 1 2 3 4 5
milk
beans
curry
bread
coffee
      products MatrixSortedDownBy 5 1 3 4 2
coffee
beans
curry
milk
bread
```

## Task 10

The 3D array `rain` gives the monthly rainfall in millimetres over 7 years for 5 countries.

```
      ⎕RL←42
      rain ←?7 5 12⍴250
```

Write a function to find the average monthly rainfall for each individual month in each of the 5 countries.

```
      ⌊ MonthAvg rain
117 137 125 106 130 133 172 118  91 140 133 113
116 146 102 147 105  73 111 138 158 128 144 126
124 106 126 101 172 126 182 109 174 126  59 135
109 121 192 138 100 131  68 156 123 140 110 159
121 120 138 147  75 132 111 102 118 117 157 109
```

## Task 11

Assign scalar numeric values (single numbers) to the variables `years`, `countries` and `months` such that the rain data can be summarised as follows:

```
    ⍴+/[years]rain          ⍝ Sum over years
5 12
    ⍴+/[countries]rain      ⍝ Sum over countries
7 12
    ⍴⌈/[months]rain         ⍝ Max over month
7 5
```

## Task 12

Write a function to find the average over an axis specified by a character scalar ⍺, with `'Y'` representing `years`, `'C'` representing `countries`, and `'M'` representing `months`.

```
      ⍴ 'Y' Avg rain   ⍝ Average over years
5 12

      ⌊0.5+ 'C' Avg rain   ⍝ Average over countries
 76 142 122 132 126 123 151 152  94  93  25 109
154 112 126 146  75 128 135 122 122  97 131 137
109 138  97 177 139  87 151 151 179 116 165 142
167 117 202 157 170 101 117  76 112 110 121 131
138 171 141  87  76 115  76 116 109 172 115 106
 83  85 129  73 102 183  93  85 154 125 163  89
 98 118 142 125 131  99 182 173 163 202 127 189

     ⍴ 'M' Avg rain   ⍝ Average over months
7 5
```

## Task 13 (bonus)

Define the following arrays in your workspace:

```
fruits ← 4 7⍴'Apples MangoesOrangesBananas'
days ← 7 3⍴'SunMonTueWedThuFriSat'
names ← 3 7⍴'Adam   RodrigoRich   '
⎕rl ← 42 1 ⋄ ate ← ?3 4 7⍴3
```

## Task 14 (bonus)

Write a function to determine who ate the most throughout the week, when only counting fruits listed in the character matrix ⍵.

```
      WhoAteMost 1 7⍴'Bananas'
Rich
      WhoAteMost 3 7⍴'OrangesBananasMangoes'
Adam
      WhoAteMost 1 7⍴'Mangoes'
Adam
Rich
```

## Task 15 (bonus)

Write a function `AteMostOnWeekdays` to determine who ate the most fruit ⍵ on weekdays ⍺:

```
      (2 3⍴'WedTue') AteMostOnWeekdays 1 7⍴'Mangoes'
Rodrigo

      (3 3⍴'MonWedFri') AteMostOnWeekdays 3 7⍴'OrangesBananasMangoes'
Adam

      (3 3⍴'MonThuFri') AteMostOnWeekdays 2 7⍴'BananasMangoes'
Rich
```

## Task 16 (bonus)

Write a function `DayMostFruitEaten` to determine the day on which people with names given in matrix ⍺ ate most of fruits given in matrix ⍵:

```
      (2 7⍴'RodrigoRich   ') DayMostFruitEaten 3 7⍴'OrangesMangoesBananas'
Tue
Thu

      (2 7⍴'Adam   Rich   ') DayMostFruitEaten 3 7⍴'OrangesApples '
Sun

      (1 7⍴'Adam   ') DayMostFruitEaten 3 7⍴'Mangoes'
Mon
Tue
Thu
Fri
```