2015 International APL Problem Solving Competition – Phase I

Phase I Tips

- We have provided you with several test cases for each problem to help you validate your solution.
- We recommend that you build your solution using dfns. A dfn (direct function) is one or more APL statements enclosed in braces {}. The left hand argument, if any, is represented in a dfn by ⍺, while the right hand argument is represented by ⍵.
  Example:
  ```
        'Hello' {⍺,'-',⍵,'!'} 'world'
  Hello-world!
  ```
  A dfn terminates on the first statement that is not an assignment. If that statement produces a result, the dfn returns that value as its result.
  Example:
  ```
        'left' { ⍵ ⋄ ⍺ } 'right'
  right
  ```
  For more information on dfns, use the online help included with Dyalog or see page 152 in http://www.dyalog.com/MasteringDyalogAPL/MasteringDyalogAPL.pdf.
- The symbol ⍝ is the APL comment symbol. In some of the examples below, comments are provided to give more information.
- Some of the problem test cases use "boxed display" to make the structure of the returned results clearer. Boxing is enabled by default on www.TryAPL.org and can be enabled in Dyalog version 14.0 and later using the user command:
  ```
        ]box on
  ```
  Without boxed display enabled:
  ```
        ⍳¨⍳4
  1  1 2  1 2 3  1 2 3 4
  ```
  With boxed display enabled:
  ```
        ⍳¨⍳4
  ```
1	1 2	1 2 3	1 2 3 4

Phase I Problems

Sample Problem - I'd like to buy a vowel
Write a dfn to count the number of vowels in a character vector.

When passed the character vector 'APL Is Cool', your solution should return:
```
4
```

Below are 2 sample solutions. Both produce the correct answer, however the first solution would be ranked higher by the competition judging committee as it demonstrates better use of array oriented programming.

```
      {+/⍵∈'AEIOUaeiou'}'APL Is Cool' ⍝ better solution
4
      {(+/⍵='A')+(+/⍵='E')+(+/⍵='I')+(+/⍵='O')+(+/⍵='U')+(+/⍵='a')+
(+/⍵='e')+(+/⍵='I')+(+/⍵='o')+(+/⍵='u')}'APL Is Cool' ⍝ lesser solution
4
```

Problem 1 – Statistics - Mean
Write a dfn that takes a numeric array as its right argument and returns the mean (average) of the array.

Test cases:

```
      {your_solution} 1 2 3 4 5 6
3.5

      {your_solution} ⍬  ⍝ the average of an empty vector is 0
0

      {your_solution} 17 ⍝ your solution should work with a scalar argument
17
```

Problem 2 – Statistics - Median
Write a dfn that takes a numeric array as its right argument and returns the median of the array. The median is the number separating the higher half of the vector from the lower half. The median can be found by arranging all the observations from lowest value to highest value and picking the middle one. If there is an even number of observations, then there is no single middle value; the median is then usually defined to be the mean of the two middle values.

Test cases:

```
      {your_solution} 1 2 3 4 5 6 7 8 9
5

      {your_solution} 1 8 2 7 3 6 4 5
4.5

      {your_solution} ⍬
0

      {your_solution} 7
7
```

Problem 3 – Statistics - Mode
Write a dfn that takes a numeric vector or scalar as its right argument and returns the mode (that is, the most common value) of the array. If more than one number occurs the greatest number of times, return all such numbers.

Test cases:

```
      {your_solution} 2 1 4 3 2 5 1 2
2

      {your_solution} ⍬ ⍝ should return an empty vector


      {your_solution} 1 2 3 4 1 2
1 2
```

Problem 4 – Just Meshing Around

Write a dfn that takes vectors as its left and right arguments and returns them "meshed" into a single vector formed by alternately taking successive elements from each argument. The arguments do not have to be the same length.

Test cases:

```
      'MENS' {your_solution} 'EKES'
MEEKNESS

      'Dyalog' {your_solution} 'APL'
DAaPlLog

      'APL' {your_solution} 'Dyalog'
ADPyLalog

       1 3 5 7 {your_solution} 2 4 6 8  ⍝ should work with numeric vectors
1 2 3 4 5 6 7 8

      '' {your_solution} 'Hello'  ⍝ either or both arguments could be empty
Hello
```

Problem 5 – You're Unique, Just Like Everyone Else

Write a dfn that takes a vector as its right argument and returns elements that occur only once in the vector.

Test cases:

```
      {your_solution} 1 2 3 4 5
1 2 3 4 5

      {your_solution} 1 2 3 4 5 4 3 2 1
5

      {your_solution} 'hello world'
he wrd
```

Problem 6 – Shorter Ones to the Front

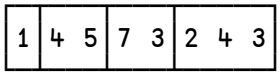
Write a dfn that takes a vector of vectors as its right argument and returns it sorted by the length of each element. Note: an element of the vector can be scalar or an empty vector.

Test cases:

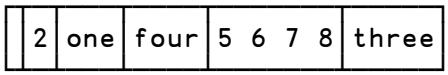
```
      {your_solution} 'one' 'two' 'three' 'four' 'five' 'six'
```

one	two	six	four	five	three

```
      {your_solution} (2 4 3) (4 5) 1 (7 3)
```




```
      {your_solution} ⍬   ⍝ should return an empty vector

      {your_solution} 'one' 2 'three' '' 'four' (5 6 7 8)
```



Problem 7 – 3s and 5s

Write a dfn that takes a numeric vector and returns all elements that are divisible by 3 or 5.

Test cases:

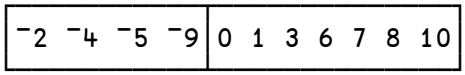
```
      {your_solution} 1 2 3 4 5 6 7 8 9 10
3 5 6 9 10

      {your_solution} ⍬   ⍝ should return an empty vector
```

Problem 8 – Separating Out the Negative

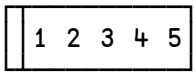
Write a dfn that takes a numeric vector and returns a two element vector whose first element contains the values less than 0 (zero) in the vector and the second element contains all values greater than or equal to 0.

Test cases:

```
      {your_solution} 0 1 ¯2 3 ¯4 ¯5 6 7 8 ¯9 10
```




```
      {your_solution} 1 2 3 4 5
```




```
      {your_solution} ⍬   ⍝ should return a vector of two empty vectors
```

Problem 9 – Delimited Text

It's common to encounter delimited text – for example, comma- separated values in a file.

Write a dfn that takes a character vector as its right argument and one or more characters as its left argument, where those characters are delimiters in the right argument. The dfn should return the delimited text as a vector of vectors.

Test cases:

```
      ',' {your_solution} 'comma,delimited,values'
```

comma	delimited	values

```
      ' ' {your_solution} 'break up words'
```

break	up	words

```
      ',' {your_solution} ','
```


Problem 10 – Order Total / Dot Product

Suppose you have a numeric vector that is the list of prices for a set of retail products. You also have a numeric vector that is the number ordered of each product. Write a dfn that takes as its right argument a vector of prices and as its left argument a numeric vector that indicates the number ordered and returns the total cost for the order.

In case you hadn't realized it, this is an application the dot product.
The dot product of two vectors **A** = [A_1, A_2, ..., A_n] and **B** = [B_1, B_2, ..., B_n] is defined as

$$A \cdot B = \sum_{i=1}^{n} A_i B_i = A_1 B_1 + A_2 B_2 + \cdots + A_n B_n$$

Test cases:

```
      5 0 2 {your_solution} 2.99 4.99 1.99
18.93

      0 0 0 {your_solution} 2.99 4.99 1.99
0
```