2014 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 `⍵`. For example:

```
      'Hello' {⍺,'-',⍵,'!'} 'world'
Hello-world!
```

The result of a dfn is the value of the first result producing statement. For example:

```
      'left' { ⍵ ⋄ ⍺ } 'right'
right
```

For more information on dfns, see page 152 in *Mastering Dyalog APL* or use the online help included with Dyalog APL.

NOTE: The symbol `⍝` is the APL comment symbol. In some of the examples below, comments are provided to give more information.

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 - It's all right

Write a dfn that takes the length of the legs of a triangle as its left argument, and the length of the hypotenuse as its right argument and returns 1 if the triangle is a right triangle, 0 otherwise.

Test cases:

```
    3 4 {your_solution} 5
1

    2 3 {your_solution} 4
0
```

Problem 2 - How tweet it is

Twitter messages have a 140 character limit; what if the limit was even shorter? One way to shorten the message yet retain most readability is to remove interior vowels from its words. Write a dfn which takes a character vector and removes the interior vowels from each word.

Test cases:

```
    {your_solution} 'if you can read this, it worked!'
if yu cn rd ths, it wrkd!

    {your_solution} 'APL is REALLY cool'
APL is RLLY cl

    {your_solution} ''   ⍝ an empty vector arg should return an empty vector

    {your_solution} 'a'  ⍝ should work with a single character message

a
```

Problem 3 - Tell a Fib

Write a dfn that takes an integer right argument and returns that number of terms in the Fibonacci sequence.

Test cases:

```
    {your_solution} 10
1 1 2 3 5 8 13 21 34 55

    {your_solution} 1
1

    {your_solution} 0   ⍝ should return an empty vector
```

Problem 4 - Space - the final frontier

Write a dfn that removes extraneous (leading, trailing, and multiple) spaces from a character vector.

Test cases:

```
    {your_solution} '  this  is a   test  '
this is a test

    {your_solution} ''   ⍝ should return an empty vector

    {your_solution} 'hello world!'
hello world!

    {your_solution} '  ' ⍝ vector of only spaces should return empty vector
```

Problem 5 - Mirror Mirror

A palindrome is a word or phrase whose letters read the same forwards and backwards. Write a dfn which returns a 1 if its character vector argument is a palindrome, 0 otherwise. For simplicity's sake, you may assume that the vector is all one case.

Test cases:

```
    {your_solution} 'a man, a plan, a canal, panama!'
1

    {your_solution} ''  ⍝ a phrase of 0 length is a palindrome
1

    {your_solution} 'a' ⍝ as is a single letter phrase
1

    {your_solution} 'APL' ⍝ APL may be cool, but it's not a palindrome
0
```

Problem 6 - Roll the dice

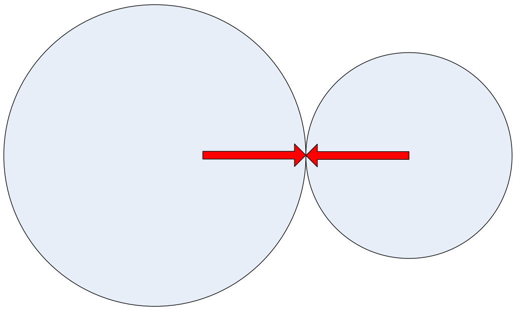
Write a dfn that takes an integer vector representing the sides of a number of dice and returns a 2 column matrix of the number of ways each possible total of the dice can be rolled.

Test cases:

```
    {your_solution} 6 6 ⍝ 2 six-sided dice
 2 1
 3 2
 4 3
 5 4
 6 5
 7 6
 8 5
 9 4
10 3
11 2
12 1

    {your_solution} 6 4 ⍝ a six-sided and a four-sided die
 2 1
 3 2
 4 3
 5 4
 6 4
 7 4
 8 3
 9 2
10 1

     {your_solution} 3  ⍝ a single 3-sided die
1 1
2 1
3 1

    {your_solution} ⍳0 ⍝ should return a matrix of shape 0 2
```

Problem 7 - Revolutionary thinking

Imagine there are two circles that are tangent to one another. One circle is stationary, the other can "roll" around the stationary circle.



Write a dfn which takes the diameters of the stationary and mobile circles and returns the number of revolutions the mobile must traverse until the tangent points meet again.

Test cases:

```
    10 {your_solution} 10 ⍝ identically sized circles
1

    10 {your_solution} 5  ⍝ a mobile circle that's half the size needs to
make 2 revolutions

2

    5 {your_solution} 7   ⍝ a mobile circle of diameter 7 needs to make 5
revolutions around a stationary circle of diameter 5

5
```

Problem 8 - Go the distance

Write a dfn that returns the distance between two points in a space of any number of dimensions.

Test cases:

```
    2 {your_solution} 5 ⍝ one-dimensional space
3
```

```
      2 2 {your_solution} 5 6 ⍝ two-dimensional space
5

      ⍬ {your_solution} ⍬ ⍝ zero dimension space
0

      2 2 3 4 {your_solution} 3 7 10 9 ⍝ four-dimensions
10
```

Problem 9 - Going ballistic

The following formula gives the horizontal distance a projectile travels:

$$distance = \frac{v^2 \sin 2\theta}{G}$$

Where: v is the initial velocity
θ is the trajectory in degrees
G is the gravitational constant

Write a dfn which calculates the distance (in meters) a projectile travels given an initial velocity in meters per second and a trajectory in degrees. Use 9.8 meters per second squared as the gravitational constant.

Test cases:

```
      100 {your_solution} 45 ⍝ 100 meters per second and 45 degree trajectory
1020.408163

       0 {your_solution} 45  ⍝ no velocity = no distance
0

       100 {your_solution} 90 ⍝ shooting straight up = no distance
1.249639591E¯13
```

Problem 10 - Sales are up, aren't they?

Given a vector representing monthly sales figures, write a dfn that returns the greatest percent month to month increase.

Test cases:

```
      {your_solution} 80 100 120 140
25

      {your_solution} 123 123 123
0

      {your_solution} 101 102 114 117 101 110 102 111 118 115 124 122
11.76470588

      {your_solution} 200 180 160 140 120
¯10
```