# Tartalomjegyzék



# Problem 1: Parenthesis Depth

Write a function called parenthesis_depth : its input is a string which is a well formed expression with parenthesis and its output is a string with same length but the symbols are changed to numbers which are the depth of the surrounding parenthesis. Don't modify the parenthesis symbols but every other symbol.

There won't be more then nine parenthesis in depth!

Example

```
a*(b+c) ? 00(111)
-(1/(2-3)) ? 0(11(222))
```

# Problem 2: Descartes Product

Write a function called descartes_product with input variadic number of sets, each set is an iterable object without repetition and its output is the Cartesian product of the input sets. If the input is n sets then the output should be a list of n-tuples!

The order of the output is not relevant but the order of the individual tuples is relevant. The tuples should follow the same order as the input sets: First element from the first set, second element from the second set ...

If there is even a single empty set in the input, then the output should be an empty list.

Examples

```
[1,2,3] ? [(1,), (2,), (3,)]
[1,2], ['a','b'] ? [(1,'a'), (1,'b'), (2,'a'), (2,'b')]
But this is not correct: [('a', 1), ...]
[1,2], ['a','b'], [3,4,5] ? [(1,'a',3), (1,'a', 4), ...]
[1,2], [], ['a','b'] ? []
```

Hint: First make it work for zero or one input set. Then apply recursion: take the product of n-1 sets and make something that produces the n-product.

Bonus: If the input sets have sizes: n1, n2 ... nk then the output should have n1*n2 ... *nk tuples.

# Problem 3: Matrix Log

Write a function with two input variables:

```
n: number of dimensions
M: a n-dimensional array as a list
```

An n-dimensional array means that it a list where each element is a (n-1)-dimensional array. A 1-dimensional array is just a list of numbers.

We would like to take the base 2 logarithms of every number in the array, but only with 2 digit precision. Return the array with the same shape as the original, where every element is the logarithm of the original element.

Hint:

```
You can use math.log() for logarithm.
You can use the built-in round() function.
Try with a recursive solution.
```

There are two ways of returning the result: overwrite the original elements in-place, using the fact that a list is mutable or return a copy of the input with the modified elements. Both can work, but try not to mix the two solutions.

# Problem 4: Target

Write a function whose input is a string of digits and an integer. Write down all possible combinations of digits and +, -, * operations that result in the given integer.

Example:

```
Target("http://wiki.math.bme.hu123"http://wiki.math.bme.hu, 6) -> '1+2+3', '1*2*3'
```

# Problem 5: New Tree Methods

- Write an `is_path(self)` method for the class Tree which returns True if the tree is a long path without a junction. False otherwise. The tree is a path if all of the nodes **except the root** have at most one branches.
- Write a `path(self)` method for the class Tree which returns all paths from the leaves to the root of a given binary tree.

# Problem 6: Proper Calculator

Write modify the calculator in the exercise to handle parantheses such that it calculates the input based on the parentheses.

Example:

```
2+(7*3) --> 23
(2+7)*3 --> 27
```

# Problem 7: Calculator Function

You have to improve the calculator's Node class even further. You have to treat the functions:

- sin, cos

Example:

```
cos(0)*sin(5+6*3)
```

- factorial: !

example:

```
(2*3)!
```

You can treat these functions as unary operations and the operator is the name of the function. This means that in the expression tree you can have "http://wiki.math.bme.husin"http://wiki.math.bme.hu, "http://wiki.math.bme.hucos"http://wiki.math.bme.hu or "http://wiki.math.bme.hu!"http://wiki.math.bme.hu in a node an they have only one outgoing edge (child) in the tree, that is their operands.

The nodes "http://wiki.math.bme.husin"http://wiki.math.bme.hu and "http://wiki.math.bme.hucos"http://wiki.math.bme.hu should have a right member, which is expression what is inside the function. This is because the argument of the sin (or cos) is on the right-hand-side of the symbol itself.

Example "http://wiki.math.bme.hu1+cos(5-3)"http://wiki.math.bme.hu:

```
 "http://wiki.math.bme.hu+"http://wiki.math.bme.hu
 / \
1  "http://wiki.math.bme.hucos"http://wiki.math.bme.hu
      \
       "http://wiki.math.bme.hu-"http://wiki.math.bme.hu
       / \
      5   3
```

The "http://wiki.math.bme.hu!"http://wiki.math.bme.hu node should have a left member, which is the argument of the factorial. This is because the argument is on the left-hand-side of the factorial symbol.

Example "http://wiki.math.bme.hu1/(2*3)!"http://wiki.math.bme.hu:

```
        "http://wiki.math.bme.hu/"http://wiki.math.bme.hu
       /  \
      1   "http://wiki.math.bme.hu!"http://wiki.math.bme.hu
          /
        "http://wiki.math.bme.hu*"http://wiki.math.bme.hu
        / \
       2   3
```

The trigonometric functions should have a higher precedence, than any other operations. Example:

```
cos(1)^2 = (cos(1))^2
```

The factorial should have a higher precedence than the binary operations, but lower than the trigonometric functions. Example:

```
1 + 2! = 1 + (2!)
```

and

```
cos(0)! = (cos(0))!
```

Help:

- The rfind method can search for a sub-string in a string.
- Use the math library's mathematical functions.

Remark:

- The argument of the factorial will be a non-negative integer, but there is a continuous generalizations of the factorial, called gamma function. "http://wiki.math.bme.hu