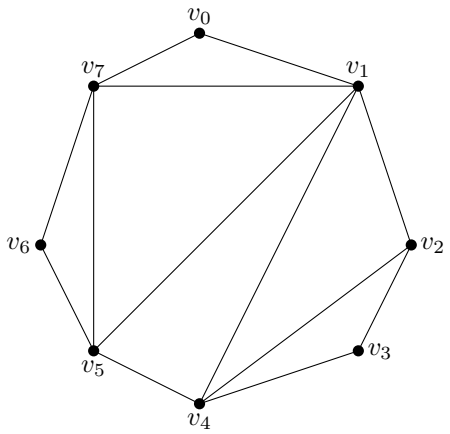
Dynamic programming – Exercises
ENSEA/FAME Computer Science

Exercise 1 – Triangulation of a convex polygon

Consider a convex polygon P whose vertices are $v_0, \ldots, v_{n-1}$. A triangulation of P is a decomposition of P into a set of non-overlapping triangles.



A polygon is represented by a list of of its vertices. For example, the above polygon is

```
>> polygon = [(0,3), (3,2), (4,-1), (3,-3), (0,-4), (-2,-3), (-3,-1), (-2,2)]
```

Question 1. Prove that a triangulation of a poylgon P involves necessarily $n-2$ triangles and $n-3$ line segments.

To each possible triangle, with vertices v_i, v_j, v_k, we associate a weight $w(i,j,k)$. A triangulation is called *optimal* if the sum of the weights of its triangles is minimal. An example of weight is the perimeter function.

Question 2. Create a function `def weight_perimeter(i, j, k, polygon)` which returns the perimeter of the triangle $v_i v_j v_k$.

In the next question, we elaborate an algorithm which returns the total weight of an optimal triangulation.

Question 3. For $0 \leqslant i < j < n$, call $t(i,j)$ the total weight of an optimal triangulation of the polygon $v_i v_{i+1} \cdots v_j$. Find a recursive formula for $t(i,j)$. Deduce an algorithm. What is its complexity?

Question 4. Suppose that the weight function is arbitrary, justify that the complexity of the previous algorithm is optimal.

Question 5. If the weight of a triangle is equal to its area, what do you think of the preceding algorithm?

Exercise 2

Matrices are implemented as lists of lists.

```
>> A = [[1, 0, 0, 0, 0, 0, 0, 1, 0],
        [1, 0, 1, 1, 1, 1, 1, 1, 0],
        [1, 0, 1, 1, 1, 1, 0, 0, 0],
        [0, 1, 1, 1, 1, 1, 0, 1, 0],
        [0, 0, 1, 1, 1, 1, 0, 0, 1],
        [0, 0, 1, 1, 1, 0, 0, 1, 1],
        [0, 1, 0, 0, 0, 0, 0, 1, 1]]
```

Find an algorithm which returns the length and position of the a maximal subsquare matrix consisting entirely of ones. For example:

```
>> maximal_subsquare(A)
(4, (1, 2))
```

(4 is the length of the maximal subsquare which starts at position $(1, 2)$.)

Exercise 3

A set $G = \{a, b, c\}$ is given an operator $\cdot$, whose table is:

	a	b	c
a	b	b	a
b	c	b	a
c	a	c	c

Note that this operation is neither associative, nor commutative. Hence, the value of a "product" *abbcabca* depends on the order of the operations. For example,

$$(((ab)(b(ca)))((bc)a) = (b(ba))(aa) = (bc)b = ab = b.$$

Question 1. Write a function `def can_be(s, letter)` which tells whether an expression `s` can be equal to `letter`. For example

```
>> can_be('abbcabca', 'a')
True
```

Question 2. What is the complexity of this function?

Exercise 4 – Longest subchain problem

Write a function `def lsc(a, b)` which returns the longest common subchain of a and b.

```
>> lsc('abccababab', 'baabcb')
'abc'
```