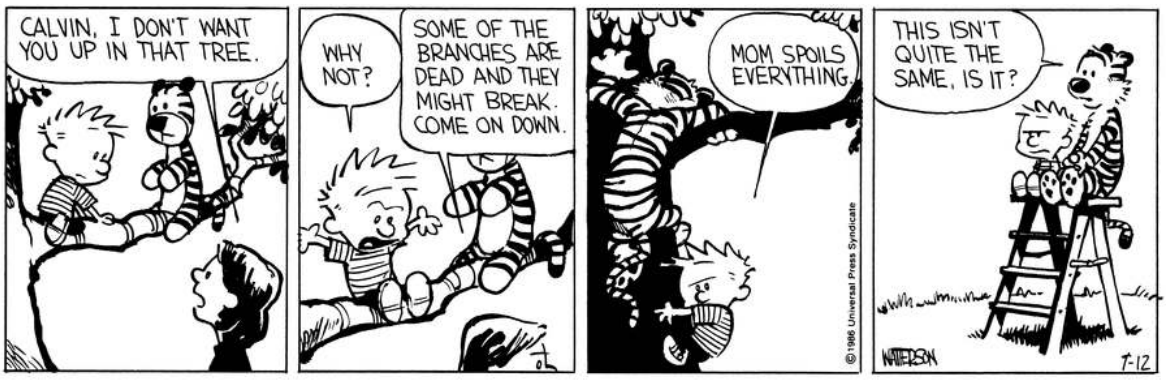
Algorithmics
Midterm #2 (C2)

Undergraduate 1^{st} year S2
EPITA

4 March 2019 - 9 : 00

Instructions (read it) :

- ☐ You must answer on **the answer sheets provided.**
  - No other sheet will be picked up. Keep your rough drafts.
  - Answer within the provided space. **Answers outside will not be marked**: Use your drafts!
  - Do not separate the sheets unless they can be re-stapled before handing in.
  - Penciled answers will not be marked.
- ☐ The presentation is negatively marked, which means that you are marked out of 20 points and the presentation points (maximum of 2) are taken off this grade.
- ☐ **Code:**
  - All code must be written in the language `Python` (no C, CAML, ALGO or anything else).
  - **Any `Python` code not indented will not be marked.**
  - All that you need (functions, methods) is indicated in the **appendix** (last page)!
- ☐ Duration : 2h



Exercise 1 (A little coursework... – *4 points*)

Let the tree `B = {`ε`,0,1,01,11,010,011,110,0101,1100,1101}`.

1. Measures:
   (a) What is the size of the tree `B`?
   (b) What is the height of the tree `B`?
   (c) What is the path length of the tree `B`?
   (d) What is the external average depth of the tree `B`?
2. Using the hierarchical numbering, give, in order, the list of nodes of the tree `B`.

Exercise 2 (Maximum gap - *4 points*)

In this exercise, the *gap* of a list is defined as the maximum difference between two values of the list. For instance, in the matrix below, the *gap* of the first line is 13.

Write the function that returns the maximum gap of the lines of a matrix (assumed non empty).

Example of result with the matrice ***Mat1*** *on the right:*

```
>>> maxgap(Mat1)
19
```

1	10	3	0	-3	2	8
-1	0	1	8	5	0	-4
10	9	14	1	4	-5	1
10	-3	7	11	6	3	0
7	8	-5	1	5	4	10

`Mat1`

Indeed the maximum gap is the one of the middle line (19 = 14 - (-5)).

Exercise 3 (Research – *4 points*)

Write the function `searchMatrix(M, x)` that returns the position (i, j) of the first value `x` found in the non empty matrix `M`. If `x` is not present, the function returns $(-1, -1)$.

Examples of results with the matrix ***Mat1*** *of the previous exercise:*

```
>>>  searchMatrix(Mat1, -5)
(2, 5)
>>> searchMatrix(Mat1, 5)
(1, 4)
>>> searchMatrix(Mat1, 15)
(-1, -1)
```

Exercise 4 (Tests – *4 points*)

Write the function `equal(`$B1$`, `$B2$`)` that tests whether the two binary trees $B1$ and $B2$ are identical. That is, if they contain the same values in the same nodes.

Exercise 5 (Leaves – *2 points*)

Write the function `leaves(`B`)` that computes the number of leaves in the binary tree B.

Exercise 6 (Mystery – *3 points*)

The function `mystery` below builds a binary tree from a list.

```
def build(L, a, b):
    if a > b:
        return None
    else:
        c = (b - a) // 2 + a
        return BinTree(L[c], build(L, a, c-1), build(L, c+1, b))

def mystery(L):
    return build(L, 0, len(L)-1)
```

1. Draw the tree that results of the application of `mystery` to the following list;
   `L = [4, 8, 2, 9, 5, 10, 1, 6, 11, 3, 12, 7, 13]`
2. What properties must have the list so that the result is:
   (a) a binary search tree (BST)?
   (b) a perfect tree?

Appendix

Binary Trees

The binary trees we work on are the same as the ones in tutorials.

- `None` is the empty tree.
- The non-empty tree is an object of class `BinTree` which has 3 attributes: `key`, `left`, `right`.

```
class BinTree:
    def __init__(self, key, left, right):
        self.key = key
        self.left = left
        self.right = right
```

Authorised functions and methods

On lists:

- `len`

Others:

- `range`
- `abs`
- `min` and `max`, but only with two integer values!

Your functions

You can write your own functions as long as they are documented (we have to known what they do).

In any case, the last written function should be the one which answers the question.