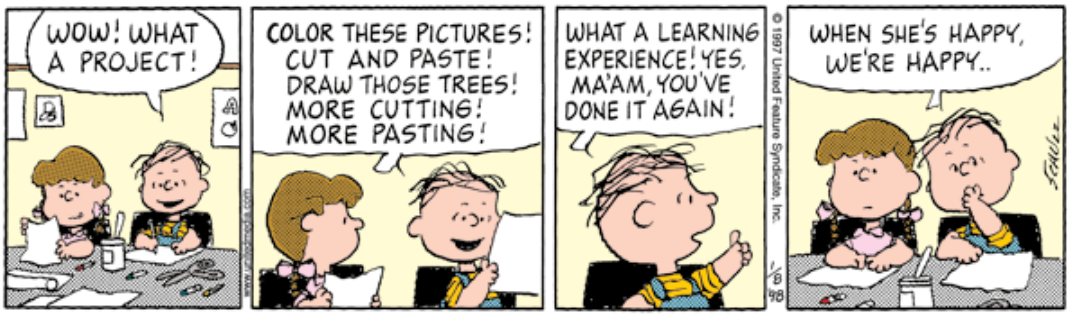
Algorithmique
Contrôle n° 2 (C2)

Info-sup S2#
Epita

8 novembre 2020 - 8 : 30

Consignes (à lire) :

- ☐ Vous devez répondre sur **les feuilles de réponses prévues à cet effet**.
  - — Aucune autre feuille ne sera ramassée (gardez vos brouillons pour vous).
  - — Répondez dans les espaces prévus, **les réponses en dehors ne seront pas corrigées** : utilisez des brouillons !
  - — Ne séparez pas les feuilles à moins de pouvoir les ré-agrafer pour les rendre.
  - — Aucune réponse au crayon de papier ne sera corrigée.
- ☐ La présentation est notée en moins, c'est à dire que vous êtes noté sur 20 et que les points de présentation (2 au maximum) sont retirés de cette note.
- ☐ **Le code :**
  - — Tout code doit être écrit dans le langage `Python` (pas de C, Caml, Algo ou autre).
  - — **Tout code `Python` non indenté ne sera pas corrigé.**
  - — Tout ce dont vous avez besoin (fonctions, méthodes) est indiqué en **annexe** !
- ☐ Durée : 2h00



Des matrices

Exercice 1 (Maximum gap - *4 points*)

Pour cet exercice, on définit le *gap* (écart) d'une liste comme étant l'écart maximum entre deux valeurs de la liste. Par exemple, dans la matrice ci-dessous le *gap* de la première ligne est 13 $(10 - (-3))$, celui de la deuxième ligne est 12 $(8 - (-4))$...

Écrire la fonction `maxgap` qui retourne le **gap maximum des lignes** d'une matrice non vide.

Exemple d'application avec la matrice `Mat1` ci-contre :

```
>>> 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`

En effet le gap maximum est celui de la ligne du milieu (19 = 14 - (-5)).

Exercice 2 (Symétrique - *4 points*)

La matrice transposée d'une matrice est la matrice A^T, obtenue en échangeant les lignes et les colonnes de A.

$$A = \begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix} \text{ alors } A^\mathsf{T} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix}$$

Une matrice *symétrique* est une matrice carrée (de taille $n \times n$) qui est égale à sa propre transposée.

Exemples :

Matrice symétrique :

	0	1	2
0	**2**	3	0
1	3	**10**	4
2	0	4	**1**

Matrice non symétrique :

	0	1	2
0	**2**	3	0
1	4	**10**	4
2	0	3	**1**

Écrire la fonction `symmetric` qui teste si une matrice carrée non vide est symétrique.

Des arbres

Exercice 3 (Représentations et questions ... – *3 points*)

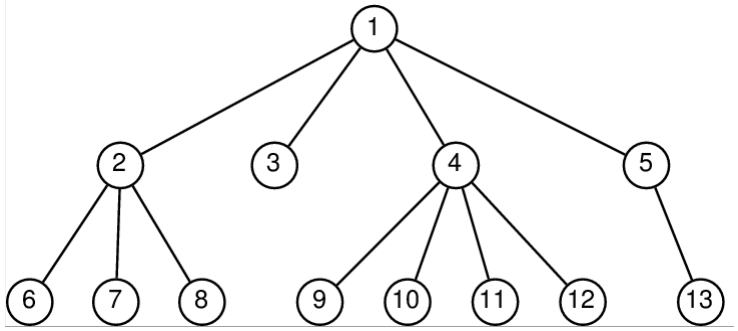


FIGURE 1 – Arbre général

1. Comment s'appelle la représentation d'un arbre général par un arbre binaire ?
2. Comment calcule-t-on la taille de l'arbre général à l'aide de l'arbre binaire le représentant ?
3. Comment calcule-t-on la hauteur de l'arbre général à l'aide de l'arbre binaire le représentant ?
4. Représenter l'arbre général de la figure 1 selon cette représentation binaire.

Des expressions et des arbres

On peut représenter une expression par un arbre : les nœuds internes contiennent les opérateurs, les opérandes, elles, sont contenues dans les nœuds externes. Les expressions (et donc les arbres) dont il est question dans les 3 prochains exercices sont des expressions arithmétiques **non vides** utilisant **uniquement les opérateurs binaires** + - * et /.

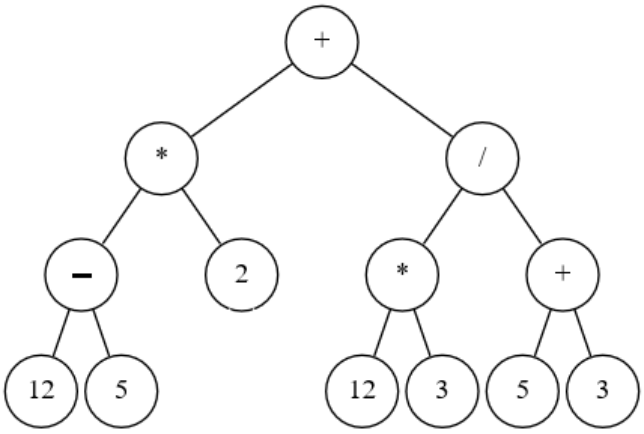


FIGURE 2 – B : Arbre de l'expression $(12-5)*2+(12*3)/(5+3)$

Exercice 4 (Dessine moi – *4 points*)

Les arbres binaires B_1, B_2 et B_3 représentent des expressions.

Lors du parcours profondeur main gauche

- les valeurs de B_1 sont rencontrées en préfixe dans cet ordre : + - / 8 2 5 + 4 * 2 3
- les valeurs de B_2 sont rencontrées en suffixe dans cet ordre : 8 20 - 2 2 * 2 + /
- Les valeurs de B_3 sont rencontrées en infixe dans cet ordre : 8 + 7 / 5 - 4 * 2
- la valeur de l'expression représentée par B_3 est -5.

Dessiner les arbres B_1, B_2, B_3 (deux solutions pour B_3, une seule demandée) et donner les valeurs des expressions représentées par B_1 et B_2.

Exercice 5 (Compte moi – *3 points*)

Écrire la fonction `nodes` qui retourne le nombre d'opérateurs, ainsi que le nombre d'opérandes d'une expression représentée par un arbre binaire non vide.

Exemple d'application sur l'arbre B (figure 2) :

```
>>> nodes(B)
(6, 7)
```

Exercice 6 (Affiche moi – *2 points*)

Écrire la fonction `exp2str` qui **retourne une chaîne** contenant l'expression complètement parenthésée représentée par un arbre binaire non vide (dont les clés sont des chaînes).

Exemple d'application sur l'arbre B (figure 2) :

```
>>> exp2str(B)
'(((12-5)*2)+((12*3)/(5+3)))'
```

Annexes

Les arbres binaires

Les arbres binaires manipulés ici sont les mêmes qu'en td.

— `None` est l'arbre vide.

— L'arbre non vide est un objet de la class `BinTree` avec 3 attributs : `key`, `left`, `right`.

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

Fonctions et méthodes autorisées

Sur les listes :

— `len`

— `append`

Autres :

— `range`

— `abs`

— `min` et `max`, mais uniquement avec deux valeurs entières !

Rappel :

Pour concaténer deux chaînes :

```
>>> s1 = "Hello"
>>> s2 = "World"
>>> s = s1 + ' ' + s2
>>> s
'Hello World'
>>> s = s + '!'
>>> s
'Hello World!'
```

Vos fonctions

Vous pouvez également écrire vos propres fonctions, dans ce cas elles doivent être documentées (on doit savoir ce qu'elles font).

Dans tous les cas, la dernière fonction écrite doit être celle qui répond à la question.