ISM MPSI/PCSI, Informatique
Année 2025/2026

# TP Informatique 20

## Exercice 1

On s'intéresse à l'algorithme d'*exponentiation rapide* réalisant un calcul performant de $x^n$ avec $x$ réel et $n$ entier. Cet algorithme s'écrit très naturellement en programmation récursive en remarquant

$$x^n = \begin{cases} \left(x^{\frac{n}{2}}\right)^2 & \text{si } n \text{ pair} \\ x \times \left(x^{\frac{n-1}{2}}\right)^2 & \text{si } n \text{ impair} \end{cases}$$

Selon la parité de $n$, la quantité $\frac{n}{2}$ pour $n$ pair et $\frac{n-1}{2}$ pour $n$ impair est le quotient de la division euclidienne de $n$ par 2.

Écrire une fonction récursive `expo(x,n)` d'arguments `x` un flottant et `n` un entier qui réalise le calcul de `x**n` par exponentiation rapide.

## Exercice 2

Soient $n$ et $k$ entiers. On pose $\binom{n}{k} = 0$ pour $k > n$.

1. Déterminer une relation entre $\binom{n}{k}$ et $\binom{n-1}{k-1}$ pour $1 \leqslant k \leqslant n$.
2. En déduire un codage récursif de `binom(n,k)` puis tester la fonction.
3. Déterminer la complexité temporelle et spatiale de `binom(n,k)`.

## Exercice 3

1. Écrire une fonction `rech_dicho(elt,L,deb,fin)` d'arguments `elt` un élément, `L` une liste non vide de nombres triée, `deb` et `fin` des entiers et qui effectue récursivement une recherche dichotomique de `elt` dans `L[deb:fin+1]`. Les cas de base correspondront aux situations :
   — `fin-deb<0` ;
   — `L[milieu]==elt` avec `milieu` une variable locale à définir.
2. Écrire une fonction `rech(elt,L)` d'arguments `elt` un nombre et `L` une liste de nombres triée non vide qui amorce la recherche récursive de `elt` dans `L` et renvoie `True,ind` si `elt` est présent dans `L` avec `ind` un indice de `elt` dans `L` et `False,ind` sinon (`ind` étant un indice quelconque dans ce cas).
3. Tester la fonction `rech` avec la liste `[1,3,5,...,17,19]` et les entiers `1, 19, 11, 12, 0, 20`.
4. Modifier la fonction `rech` en commencant par un test `assert` afin de vérifier le caractère trié de la liste transmise en argument. On pourra écrire une fonction `est_triee(L)` d'argument `L` une liste non vide de nombres qui renvoie `True` si la liste est triée et `False` sinon. Tester cette nouvelle version sur des situations conformes et non conformes.

## Exercice 4

Soit $f \in \mathscr{C}^0([\,a\,;b\,],\mathbb{R})$ vérifiant $f(a)f(b) \leqslant 0$.

1. Écrire une fonction récursive `dicho(f,a,b,eps)` qui détermine par dichotomie une racine de $f$ sur $[\,a\,;b\,]$ à la précision `eps`.
2. Tester la fonction `dicho` sur les situations suivantes :
   — $f : t \mapsto t^2 - 2$ sur $[\,0\,;2\,]$ ;
   — $f : t \mapsto \sin t$ sur $[\,2\,;4\,]$.
3. Écrire une fonction `rech_dicho` commençant par un test `assert` afin de vérifier que l'hypothèse $f(a)f(b) \leqslant 0$ est vérifiée et qui lance, le cas échéant, la recherche dichotomique récursive. Tester cette nouvelle version sur des situations conformes et non conformes.

## Exercice 5

Soient $a$, $b$, $q$ et $r$ des entiers tels que $a = bq + r$. On note $a \wedge b$ le pgcd de $a$ et $b$. On rappelle la propriété d'Euclide

$$a \wedge b = b \wedge r$$

Écrire une fonction récursive `pgcd(a,b)` qui calcule le pgcd des entiers `a` et `b`.

## Exercice 6

Écrire une fonction récursive `val(n,p)` qui pour un entier $n$ détermine la plus grande puissance de $p$ factorisable dans $n$. Tester la fonction puis estimer sa complexité temporelle.

## Exercice 7

On définit $\varphi$ la *fonction indicatrice d'Euler* sur $\mathbb{N}^*$ par

$$\forall n \in \mathbb{N}^* \qquad \varphi(n) = \text{Card}\,\{k \in [\![\,1\,;\,n\,]\!] \mid k \wedge n = 1\}$$

1. Écrire une fonction `phi(n)` d'argument `n` entier qui renvoie $\varphi(n)$. La fonction pourra faire appel à la fonction récursive `pgcd`.
2. On peut montrer la relation

   $$\forall n \in \mathbb{N}^* \qquad n = \sum_{d|n} \varphi(d)$$

   où la somme s'entend au sens des diviseurs positifs de $n$. En déduire une implémentation récursive `phi_rec(n)` de la fonction indicatrice d'Euler et vérifier celle-ci en comparant ses résultats avec ceux de `phi(n)`.