TP 15 - Complexité, correction, terminaison

Exercice 1

```
def h(n):
    d=2
    while n%d!=0 and d*d<=n:
        d=d+1
    if d*d<=n:
        return False
    else:
        return True
```

1. Identifier ce que fait la fonction ci-dessus.
2. Expliquer son déroulement.
3. Préciser un variant et établir la terminaison.
4. Préciser un invariant et établir la correction.

Exercice 2

1. Proposer un algorithme qui détermine la plus petite puissance de deux supérieure (ou égale) à un entier n donné.
2. Proposer un invariant de boucle. Donner la complexité.
3. Adapter cet algorithme pour déterminer la taille binaire de n.

Exercice 3

1. Compléter l'algorithme suivant qui calcule le polynôme de Taylor de la fonction sinus

$$\sum_{k=0}^{n}(-1)^k\frac{x^{2k+1}}{(2k+1)!} \text{ avec } x\in\mathbb{R} \text{ et } n\in\mathbb{N}$$

```
s ← ...
m ← ...
f ← ...
pour k variant de 1 à n faire
  s ← s+m/f
  m ← ...
  f ← ...
finpour
afficher s
```

2. Proposer une invariant de boucle :

3. Vérifier la terminaison et la correction.

Exercice 4 *Compréhension d'un algorithme* **Prog**

Considérons l'algorithme suivant :

```
1 def algo(x,n):
2     r,m,p=1,x,n
3     while p!=0:
4         if p%2==1:
5             r=r*m
6         m,p=m*m,p//2
7     return(r)
```

1. Suivre dans un tableau le contenu des variables ou expression lors de l'exécution de `algo(-2,13)` :

r	m	p	p%2 (test lig.4)
...	...	...	$\emptyset$
⋮	⋮	⋮	⋮

La première ligne est pour l'initialisation, puis une ligne par boucle.

2. Donner l'écriture binaire de 13 et faire le lien avec le tableau.
3. Montrer que $rm^p = x^n$ est un invariant de boucle de `algo`.
4. En déduire l'expression retournée par `algo` en fonction de x et n.
5. Établir la terminaison de la fonction.
6. Évaluer la complexité temporelle de `algo`.
7. Proposer une version récursive : `algoR`.

Faire le tableau du contenu des variables ou expressions pour l'instance $(-2,13)$.

8. Combien de multiplications de matrices sont nécessaires pour calculer A^{100} avec $A\in\mathcal{M}_2(\mathbb{R})$ en utilisant l'algorithme de l'exponentiation rapide.

Mise en forme de l'exponentiation rapide :

- l'écriture binaire de n : $n=(b_p\dots b_1b_0)_2 = \sum_{k=0}^{p} b_k 2^k$
- calcul de la x^n :

$$x^n = x^{\sum_{k=0}^{p} b_k 2^k} = \prod_{k=0}^{p} x^{b_k 2^k} = \prod_{k=0}^{p}\left(x^{2^k}\right)^{b_k}$$

➤ On note que $x^{2^{k+1}} = \left(x^{2^k}\right)^2$; ainsi, la suite des valeurs de x^{2^k} se calcule par le calcul successif de carrée : $m \leftarrow m^2$.

➤ Lorsque $b_k = 1$, ce qui correspond à un reste non nul dans la suite des divisions successives de n par 2, à savoir lorsque le test p%2==1 est vrai, alors le facteur x^{2^k} est comptabilisé : $r \leftarrow r\times m$.

9. Mettre en place l'évaluation de Hörner dans l'écriture de n comme l'évaluation du polynôme $P=\sum_{k=0}^{p} b_k X^k$ en 2.

En déduire une autre façon de calculer A^{100}.

Lycée Saint-Exupery – maffre.saintexupery@gmail.com – MPSI