1) Montrons que :

$$x - \frac{x^2}{2} \leq \ln(1+x) \leq x$$

Posons

$$f(x) = x - \ln(1+x)$$

$$f'(x) = 1 - \frac{1}{x+1} = \frac{x+1-1}{x+1} = \frac{x}{x+1}$$

Sur $\mathbb{R}^+$, $f'(x) \geq 0$, donc la fonction f est croissante. Or $f(0) = 0$ donc, pour tout x, $f(x) \geq f(0) = 0$ et donc

$$\forall x \in \mathbb{R}^+ \qquad \ln(1+x) \leq x$$

De même, posons

$$g(x) = \ln(1+x) - x + \frac{x^2}{2}$$

$$g'(x) = \frac{1}{x+1} - 1 + x = \frac{1-(1-x)(1+x)}{x+1} = \frac{1-1+x^2}{x+1} = \frac{x^2}{x+1}$$

Pour tout $x \in \mathbb{R}^+$ nous avons $g'(x) \geq 0$, la fonction g est croissante et pour tour tout x $g(x) \geq g(0) = 0$, on en déduit

$$\ln(1+x) \geq x - \frac{x^2}{2}$$

Conclusion :

$$\forall x \in \mathbb{R}^+ \qquad x - \frac{x^2}{2} \leq \ln(1+x) \leq x$$

2) On pose pour $\in \mathbb{N}^\star$

$$v_n = \sum_{i=1}^{n} \ln\left(1 + \frac{i^3}{n^4}\right)$$

Montrons que :

$$0 \leq \sum_{i=1}^{n} \frac{i^3}{n^4} - v_n \leq \frac{1}{2} \sum_{i=1}^{n} \frac{i^6}{n^8}$$

D'après la question 1) nous avons

$$\frac{i^3}{n^4} - \frac{1}{2}\frac{i^6}{n^4} \leq \ln\left(1 + \frac{i^3}{n^4}\right) \leq \frac{i^3}{n^4}$$

$$-\frac{i^3}{n^4} \leq -\ln\left(1 + \frac{i^3}{n^4}\right) \leq -\frac{i^3}{n^4} + \frac{1}{2}\frac{i^6}{n^4}$$

$$0 \leq \frac{i^3}{n^4} - \ln\left(1 + \frac{i^3}{n^4}\right) \leq \frac{1}{2}\frac{i^6}{n^4}$$

En sommant, il vient

$$0 \leq \sum_{i=1}^{n} \frac{i^3}{n^4} - \sum_{i=1}^{n} \ln\left(1 + \frac{i^3}{n^4}\right) \leq \frac{1}{2} \sum_{i=1}^{n} \frac{i^6}{n^8}$$

Conclusion :

$$0 \leq \sum_{i=1}^{n} \frac{i^3}{n^4} - v_n \leq \frac{1}{2} \sum_{i=1}^{n} \frac{i^6}{n^8}$$

3) Considérons le développement de $(i+1)^7$

$$(i+1)^7 = i^7 + 7i^6 + \sum_{k=0}^{5} C_7^k i^k$$

donc

$$i^6 = \frac{1}{7}\left((i+1)^7 - i^7 - \sum_{k=0}^{5} C_7^k i^k\right)$$

Et en sommant, il vient

$$\sum_{i=1}^{n} i^6 = \frac{1}{7}\left(\sum_{i=1}^{n}(i+1)^7 - \sum_{i=1}^{n} i^7 - \sum_{i=1}^{n}\sum_{k=0}^{5} C_7^k i^k\right)$$

$$\sum_{i=1}^{n} i^6 = \frac{1}{7}\left((n+1)^7 - 1 - \sum_{i=1}^{n}\sum_{k=0}^{5} C_7^k i^k\right)$$

Autrement dit la somme des i^6 est un polynome en n de degré 7.

$$\sum_{i=1}^{n} i^6 = P(n)$$

Donc

$$\frac{1}{2}\sum_{i=1}^{n}\frac{i^6}{n^8} = \frac{1}{2n^8}\sum_{i=1}^{n} i^6 = \frac{P(n)}{2n^8}$$

$$\lim_{n\to\infty}\sum_{i=1}^{n}\frac{i^6}{n^8} = 0$$

Limite d'une fraction rationnel dans laquelle le degré du dénominateur est supérieur au degé du numérateur. Le théorème des gendarmes (ou d'encadrement) nous permet de conclure que :

$$0 \le \lim_{n\to\infty}\left(\sum_{i=1}^{n}\frac{i^3}{n^4} - v_n\right) \le \lim_{n\to\infty}\frac{1}{2}\sum_{i=1}^{n}\frac{i^6}{n^8} = 0$$

d'ou

$$\lim_{n\to\infty}\left(\sum_{i=1}^{n}\frac{i^3}{n^4} - v_n\right) = 0$$

et donc

$$\lim_{n\to\infty} v_n = \lim_{n\to\infty}\sum_{i=1}^{n}\frac{i^3}{n^4}$$

Or

$$\sum_{i=1}^{n}\frac{i^3}{n^4} = \frac{1}{n^4}\sum_{i=1}^{n} i^3 = \frac{1}{n^4}\left(\frac{n(n+1)}{2}\right)^2$$

La limite de la fraction rationnelle est immédiate.

$$\frac{1}{n^4}\left(\frac{n(n+1)}{2}\right)^2 = \frac{1}{4}\frac{n^4+2n^3+n^2}{n^4} = \frac{1}{4}\left(1+\frac{2}{n}+\frac{1}{n^2}\right)$$

donc

$$\lim_{n\to\infty}\frac{1}{n^4}\left(\frac{n(n+1)}{2}\right)^2 = \lim_{n\to\infty}\frac{1}{4}\left(1+\frac{2}{n}+\frac{1}{n^2}\right) = \frac{1}{4}$$

Conclusion :

$$\lim_{n\to\infty} v_n = \frac{1}{4}$$

4) Calculons maintentant la limite de la suite (u_n). Nous avons

$$u_n = \prod_{i=1}^{n}\left(1+\frac{i^3}{n^4}\right) \qquad \ln u_n = \sum_{i=1}^{n}\ln\left(1+\frac{i^3}{n^4}\right) = v_n$$

Donc

$$\forall n \in \mathbb{N}^\star \qquad u_n = e^{v_n}$$

$$\lim_{n\to\infty} v_n = \frac{1}{4} \implies \lim_{n\to\infty} u_n = e^{\frac{1}{4}} = \sqrt[4]{e}$$

Conclusion :

$$\lim_{n\to\infty} u_n = \sqrt[4]{e}$$

Pour les questions sur l'alogrithmie en Python voir les exemples d'algorithmes.

Pour la question 3) nous aurions pu utiliser le résultat suivant

$$\forall n \in \mathbb{N}^\star \qquad \sum_{k=1}^{n} k^3 = \left(\sum_{k=1}^{n} k\right)^2$$

On peut également montre de façon plus générale que

$$\sum_{k=1}^{n} u_k^3 = \left(\sum_{k=1}^{n} u_k\right)^2$$

5) Ecrivons le code Python d'une fonction f qui crée la liste des cubes des entiers de 1 à n

```
def f(n):                # Definition de la fonction f
    l = range(1,n+1)     # Range genere une liste de 1 a n
    return l             # Renvoi la liste
```

6) Ecrivons le code Python d'une fonction « somme »qui retourne la somme des éléments d'une liste passée en paramètre.

```
def somme(l):
    s = 0
    for t in l:          # Parcours de la liste
        s = s + t        # sommation des elements
    return s
```

7) Ecrivons le code Python d'une fonction « produit »qui retourne la somme des éléments d'une liste passée en paramètre.

```
def produit(l):
    p = 1
    for f in l:
        p = p * f
    return p
```

8) Ecrivons le code Python d'un fonction g qui calcul le n^{eme} terme de la suite (v_n)

```
def g(n):
    li = f(n)
    l = []
    for i in li:
        l.append( log(1 + float(i)**3/float(n)**4 ))
    return somme( l )
```

9) Ecrivons le code Python d'un fonction h qui calcul le n^{eme} terme de la suite (vu_n)

```
def h(n):
    li = f(n)            # Generation de la liste 1,2,...,n
    l = []               # Creation d'une liste vide
```

```
    for i in li:            # Pacours l et calcul les facteurs
        l.append(1 + float(i)**3/float(n)**4)
    return produit( l )     # Calul le produit
```

En regroupant l'ensemble des fonctions nous obtenons l'algorithme suivant calculant le 1000^{eme} terme des suites (u_n) et (v_n)

```
#!/usr/bin/python

from math import log              # Import de la fonction log

def f(n):                   # Definition de la fonction f
    l = range(1,n+1)            # Range genere une liste de 1 a n
    return l                # Renvoi la liste

def somme(l):
    s = 0
    for t in l:             # Parcours de la liste
        s = s + t           # sommation des elements
    return s                # Renvoi de la valeur

def produit(l):
    p = 1
    for f in l:
        p = p * f
    return p

def g(n):
    li = f(n)
    l = []
    for i in li:
        l.append( log(1 + float(i)**3/float(n)**4 ))
    return somme( l )

def h(n):
    li = f(n)                   # Generation de la liste 1,2,...,n
    l = []                      # Creation d'une liste vide
    for i in li:                    # Pacours l et calcul les facteurs
        l.append(1 + float(i)**3/float(n)**4)
    return produit( l )         # Calul le produit

print g( 1000 )                 # Appel g pour calcul de Vn
print h( 1000 )                 # Appel h pour calcul de Un
```

Nous pouvons légèrement le modifier afin de passer la valeur de n en argument de la ligne de commande

```
#!/usr/bin/python

import sys
from math import log

def f(n):
    l = range(1,n+1)
    return l

def somme(l):
    return sum(l)

def produit(l):
    return reduce( lambda x,y: x*y, l)

def g(n):
    l = f(n)
    l = [ log(1 + float(i)**3/float(n)**4 ) for i in l]
    return somme( l )

def h(n):
    l = f(n)
    l = [ 1 + float(i)**3/float(n)**4 for i in l]
    return produit( l )


if len( sys.argv ) < 2:
    print "Entrer n"
    sys.exit(1)

n = int(sys.argv[1])
print "Un =", g( n )
print "Vn =",  h( n )
```

Donnons enfin une version très compacte du code utilisant les fonctions anonymes et reduce.

```
#!/usr/bin/python

def u(n):
    return reduce( lambda x,y: x*y, [ 1 + i**3 / float(n)**4 for i in range( 1, n+1) ]
    )

n = int( input( "Entre la valeur de n : " ) )
print "Un=", u( n )
```