**CORRECTION D'EXAMEN**
**IN14 – Algorithmique et Programmation**
**Année universitaire 2017-2018**

| | |
|---|---|
| Classe : | **Aéro 1** |
| Type d'examen : | **Partiel** |
| Date : | 15/01/2017 |
| Durée : | **2 heures** |
| Code matière : | IN14 |
| Intitulé matière : | Algorithmique et programmation |
| Enseignants : | M.BEN THAIER, F. Bonnefoi, V. Quetin, F.Bel bechir , A. Bore |
| Examen initial : | Oui |
| Documents autorisés : | Non |
| Calculatrices autorisées : | Non |

CADRE RÉSERVÉ A L'ETUDIANT(E) :

*En cas de réponse directement sur le sujet, merci de compléter ce cadre :*

**NOM** :

**Prénom** :

**Classe** :

# Partie 1 : Algorithmique

**Cocher la ou les bonnes réponses aux questions suivantes :**

1. **Quelle est la bonne réponse pour imprimer la racine carrée de 2 en important la bibliothèque math de la manière suivante ? (1pt)**

```
#-----------------------------
import math as m
#-----------------------------
```

- ○ print (m.sqrt(2))
- ○ print (sqrt(2))
- ○ print (math.sqrt(2))
- ○ print(racine.sqrt(2))

2. **Qu'est-ce que c'est ? (1pt)**

```
#--------------------------------
def u_rec( n ) :
    if n == 0 :
        return 2
    else:
        return 3* u_rec(n-1) -1
#--------------------------------
```

- ○ C'est une procédure
- ○ C'est une fonction itérative
- ○ C'est une fonction récursive
- ○ C'est une fonction impossible
- ○ C'est une liste

3. **Que donne le mini code Python suivant ? (1pt)**

```
#--------------------------------
n = 0
while n<15 :
    n = n + 2
print(n)
#--------------------------------
```

- ○ 14
- ○ 15
- ○ 16
- ○ 17

4. **Quel est le résultat de la Fonction F (Suite de Newman-Conway) : (1pt)**

```
###----------------------------
def  F(n) :
    if ( n == 1 or n==2):
      return 1
    else:
      return F(F(n-1)) + F(n-F(n-1))
#----------------------------
#      Programme
#----------------------------
print (F(5))
###----------------------------
```

- ○ 1
- ○ 2
- ○ 3
- ○ 4
- ○ Impossible pas de solution

5. **Quel est le nombre d'appel de la fonction impair dans le programme ? (1pt)**

```
#----------------------------
def pair(n):
    if n == 0:
       return True
    else:
       return impair(n-1)
#----------------------------
def impair(n):
    if n == 0:
       return False
    else:
       return pair(n-1)
#----------------------------
#      Programme
#----------------------------
print (pair (5))
#----------------------------
```

- ○ 0
- ○ 1
- ○ 2
- ○ 3
- ○ 4
- ○ 5

6. **Comment s'appelle la méthode de la bibliothèque os de Python qui nous fait changer de répertoire de travail ? (1pt)**

- ○ mkdir
- ○ chdir
- ○ listdir

7. **Soit le fichier test.txt contenant : (1pt)**

```
une ligne
une autre ligne
une dernière ligne
en fait non
```

**Que va être affiché par Python lors du dernier `print()` dans le code suivant ?**

```
#----------------------------
i=0
f = open('test.txt', 'r+')
for l in F.readlines() :
    if("une" in l) :
        print(l)
        i=i+1
f.close()
print(i)
#----------------------------
```

- une
- en (avec un espace en fin )
- toutes les lignes contenant une
- le nombre de lignes contenant une
- une erreur, le code est incorrect

## Partie 2 : Compréhension de code

8. **Qu'affiche le code Python suivant ? (1pt)**

```
print('1 + 2')
```

| |
|---|
| 1+2 |

9. **Quelle sera la valeur de i après l'exécution du code Python suivant ? (1pt)**

```
i = 0
while i < 10:
    i = i + 2
    print(i)
```

| |
|---|
| 2<br>4<br>6<br>8<br>10 |

10. **Que donnera le print après l'exécution du code Python suivant ? (1pt)**

```
x = 'ohlala'
print(x[3:4] + x[4:] + x[:2])
```

alaoh

**11. Quel affichage est produit par le code python suivant ? (1pt)**

```
def f(x):
    return x * 2
print(f(f(2)))
```

8

**12. Quel affichage est produit par le code python suivant ? (1pt)**

```
a = [3, 1, 4, 1, 5]
b = []
for i in range(len(a) - 1):
    b.append(a[i] + a[i + 1])
print(b)
```

[4, 5, 5, 6]

**13. Quel est le dernier affichage produit par le code python suivant : (1pt)**

```
s = ''
for i in range(8):
    if i % 2 == 0:
        s = s + str(i)
    else:
        s = s + '-'
print(s+str(8))
```

0-2-4-6-8

**14. Quel affichage est produit par le code python suivant : (1pt)**

```
x = 7
def f(b):
    x = 3 * b
    print(x)
    return x + 1
print(f(x))
print(x)
```

21
22
7

# Partie 3 : Production de code

**15. Ecrire une fonction « factorielle_iterative » qui prend en paramètre un entier n >0 et retourne la factorielle de ce nombre. (1.5pt)**

```
#-------------------------------------------------------------------
#            Exercice 1
#-------------------------------------------------------------------
# Ecrire une fonction factorielle itérative ( non récursive )
# qui permet de saisir un entier n >=0 et de retourner sa factorielle
#-------------------------------------------------------------------
def factorielle_itérative(n):
    R=1
    for i in range (n,0,-1):
        R=R*i
    return(R)
```

**16. Ecrire une fonction « factorielle_recursive » qui prend en paramètre un entier n >0 et retourne la factorielle de ce nombre. (1.5pt)**

```
#-------------------------------------------------------------------
#            Exercice 2
#-------------------------------------------------------------------
# Ecrire une fonction factorielle récursive
# qui permet de saisir un entier n >=0 et de retourner sa factorielle
#-------------------------------------------------------------------
def factorielle_recursive(n):
    if( n==0 or n== 1 ):
        return(1)
    else:
        return(n*factorielle_recursive(n-1))
```

**17. Ecrire une fonction pour calculer le nombre de combinaisons de p éléments parmi n, utilise les identités suivantes (0 ≤ p ≤ n) : (1.5pt):**

$$\binom{n}{k} = C_n^k = \frac{n!}{k!\,(n-k)!}$$

```
#-------------------------------------------------------------------
#            Exercice 3
#-------------------------------------------------------------------
# Ecrire une fonction pour calculer le nombre de combinaisons
# de p éléments parmi n, utilise les identités suivantes (0 ≤ p ≤ n) :
#-------------------------------------------------------------------
```

```
def combinaison (n,p):
    return( factorielle_recursive(n))/((factorielle_recursive(p))*(factorielle_recursive(n-p)))
#---------------------------------------------------------------------
```

**18. Ecrire une procédure qui prends en paramètre le nom d'un fichier texte et permet d'enregistrer le triangle de Pascal issu du binôme de Newton dans ce fichier en utilisant la fonction combinaison. (1.5pt)**

$$(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} . a^{n-k} . b^k$$

```
#---------------------------------------------------------------------
#            Exercice 4
#---------------------------------------------------------------------
def enregistrement ( fichier , n ):
    F=open(fichier,'w')
    for i in range (0,n+1):
        R=""
        for k in range ( 0,i+1):
            R=R+(str(int(combinaison(i,k)))+"\t")
        F.write(R)
        F.write("\n")
    F.close()

#---------------------------------------------------------------------
```

Les variantes sont acceptées si elles ne sont pas trop farfelues.
Bonus de +0.25 si commentaires en début de fonction.

Exemple : En exécutant le fichier enregistrement ( "ma_combinaison.txt",6 ):
J'aurai dans mon fichier «ma_combinaison.txt » le triangle de Pascal suivant :

```
1
1       1
1       2       1
1       3       3       1
1       4       6       4       1
1       5       10      10      5       1
1       6       15      20      15      6       1
```